8-2013

Short-term Electrical Load Forecasting for an Institutional/ Industrial Power System Using an Artificial Neural Network

Eric Lynn Taylor
University of Tennessee – Knoxville., etaylo10@utk.edu

To the Graduate Council:

I am submitting herewith a thesis written by Eric Lynn Taylor entitled "Short-term Electrical Load Forecasting for an Institutional/Industrial Power System Using an Artificial Neural Network." I have examined the final electronic copy of this thesis for form and content and recommend that it be accepted in partial fulfillment of the requirements for the degree of Master of Science, with a major in Electrical Engineering.

Leon M. Tolbert, Major Professor

We have read this thesis and recommend its acceptance:

Yilu Liu, Kai Sun

Accepted for the Council:
Carolyn R. Hodges

Vice Provost and Dean of the Graduate School

(Original signatures are on file with official student records.)

Short-term Electrical Load Forecasting for an Institutional/Industrial Power System Using an Artificial Neural Network

A Thesis Presented for
the Master of Science
Degree
The University of Tennessee, Knoxville

Eric Lynn Taylor
August 2013

Acknowledgements

I would like to thank my wife, Kim, and children, Emma, Julia, and Kaitlyn, for their support and understanding while I completed my graduate studies. I thank my Graduate Committee, Dr. Leon Tolbert, Dr. Yilu Liu, and Dr. Kai Sun, for their time, guidance, and input into this research. I would like to thank Michael Starke for his assistance in helping me choose a thesis topic and for obtaining some of the data required for this research. I thank my management team at Y-12, Dave Harvey, Nick Antonas, and Sam O'Neal for their continued encouragement, interest in my graduate studies, and understanding when I needed to leave work in the middle of the day to attend class. I also want to thank all of my family and friends whose encouragement and confidence in me helped me see my studies through to the end and graduate.

Abstract

For optimal power system operation, electrical generation must follow electrical load demand. The generation, transmission, and distribution utilities require some means to forecast the electrical load so they can utilize their electrical infrastructure efficiently, securely, and economically. The short-term load forecast (STLF) represents the electric load forecast for a time interval of a few hours to a few days. This thesis will define STLF as a 24-hour-ahead load forecast whose results will provide an hourly electric load forecast in kilowatts (kW) for the future 24 hours (a 24-hour load profile).

This thesis will use the method of Artificial Neural Networks (ANN) to create a STLF algorithm for the U.S. Department of Energy's Oak Ridge National Laboratory (ORNL). ORNL's power system can be described as an institutional/industrial-type electrical load. The ANN is a mathematical tool that mimics the thought processes of the human brain. The ANN can be created and trained to receive historical load and future weather forecasts as input and produce a load forecast as its output. Most ANNs in the literature are used to forecast the next day 24-hour load profile for a transmission-level system with resulting load forecast errors ranging from approximately 1 % to 3 %. This research will show that an ANN can be used to forecast the smaller, more chaotic load profile of an institutional/industrial-type power system and results in a similar forecast error range. In addition, the operating bounds of the ORNL electric load will be analyzed along with the weather profiles for the site. Correlations between load and weather and load and calendar descriptors, such as day of week and month, will be used as predictor inputs to the ANN to optimize is size and accuracy.

Table of Contents

List of Tables

List of Figures

List of Attachments

Chapter 1
Introduction

1.1 General

At present, there is no substantial energy storage in the electric transmission and distribution system. For optimal power system operation, electrical generation must follow electrical load demand. The generation, transmission, and distribution utilities require some means to forecast the electrical load so they can utilize their electrical infrastructure efficiently, securely, and economically. Generation utilities use electrical load forecasting techniques to schedule their generation resources to meet the future load demand. Transmission utilities use electric load forecasting techniques to optimize the power flow on the transmission network to reduce congestion and overloads. Distribution utilities would not have much interest in short-term electric load forecasts as their distribution systems are predominantly radial with predictable maximum load demands. Thus, the distribution systems are sized conservatively and short-term load changes have little effect on the distribution system.

Long- and medium-term load forecasts predict the electrical load over time ranges measured in months or years. The short-term load forecast (STLF) represents the electric load forecast for a time interval of a few hours to a few days [1]. This thesis will define STLF as a 24-hour-ahead load forecast whose results will provide an hourly electric load forecast in megawatts (MW) for the future 24 hours (a 24-hour load profile).

The intent of this research is to perform short-term electric load forecasting for a specific facility, whereas most other STLF studies are for utility transmission and distribution systems. The facility to be studied is the U.S. Department of Energy's Oak Ridge National Laboratory (ORNL). ORNL's power system can be described as an institutional/industrial-type electrical load which consists of commercial office buildings, high-performance computing facilities, fabrication shops, research laboratories, and utility infrastructure such as chiller plants and a steam plant. ORNL is directly served by the Tennessee Valley Authority (TVA) at 161 kV. The voltage is stepped down to 13.8 kV at the utility substation and distributed throughout the site. During fiscal years FY11 and FY12, ORNL's average power usage was approximately 40 MW with a metered summer peak demand of approximately 51 MW.

The usefulness of STLF for specific load centers such as ORNL is that the facility can use the load forecasts and energy demand management techniques to plan for peak electrical load reduction through various means such as on-site generation, electric load-shedding, and implementing demand-response agreements with the serving utility.

ORNL falls within TVA's "Manufacturing" rate structure. This structure defines the energy rate ($/kWh) and demand rate ($/kW) for each month. This rate structure is divided into three seasons (summer, winter, and transition) each with different rate charges. To illustrate the value of peak load management, if ORNL's maximum summer demand load of 51 MW was reduced by 5 %, then there would be a 5 % reduction in the demand charge which would save hundreds of thousands of dollars each year in electricity costs.

1.2 Load Forecasting Techniques

Some of the techniques that have been proposed and implemented to create STLF are:

1. Multiple linear regression
2. Stochastic time series
3. General exponential smoothing
4. State space method
5. Knowledge-based expert approach
6. Artificial Neural Network (ANN)

Methods 1-4 use statistical means to arrive at a forecast solution [2]. The algorithm for method 5 selects a reference day based on a set of rules and reshapes this day's electric load curve using other sets of rules specific to the system under study [2]. Method 6 uses an algorithm that combines previous system load and weather data and predicts a future load pattern. The ANN is trained with an input data set to approximate a target data set. Load forecasting is an inherent nonlinear problem and the structure of an ANN is suited for nonlinear modeling [2]. These six methods will be explored further in Chapter 2 with the focus on ANN as the preferred method for calculating the STLF for an industrial/institutional facility. Previous electrical load and weather data for ORNL will be used as input and training data for an ANN. The algorithm will be optimized to produce a STLF with low percent error.

STLF has been extensively researched and modeled at the generation and transmission level, but research and modeling of STLF at the distribution and load level

is scare to non-existent. One possible reason for this lack of research is that larger utilities are concerned with generation scheduling and system peaking, not small sectors of short-term load demand. Another possible reason is that distribution utilities are not concerned with an individual customer's short-term load demand as long as the distribution system can supply the maximum load.

1.3 Outline of Thesis

Chapter 2 describes the more popular techniques for electric load forecasting. It reviews the existing literature and the various methods and focuses on two methods that will be used for creating a STLF for an institutional/industrial facility.

Chapter 3 presents the analysis techniques used to implement a STLF for ORNL. The methods and Matlab® functions used to create an ANN are discussed in detail.

Chapter 4 presents the STLF simulation results and compares the forecasted results with the actual 24-hour load. The load bounds and behavior are studied as well as ORNL's weather profiles and their effect on the load profile.

Chapter 5 summarizes the research and results presented in this thesis. This chapter also proposes future work in electrical STLF for institutional/industrial facilities.

Chapter 2

Literature Review

Predicting the electric power consumption of an individual piece of equipment in a large facility can be difficult if not impossible without specific metering data for this load. Typically, the usage of a single electrical device in a larger power system is random and usage patterns of other devices may differ from the one under study. There is often a large diversity in individual loads, yet when these individual loads are summed into one larger facility load, patterns emerge which can be statistically predicted [3].

There are four main factors that influence electrical load [3]:

1. Economic
2. Time
3. Weather
4. Random effects

2.1 Economic factors

Economic factors consist of investment in the facility's infrastructure through construction of new buildings, labs, and experiments which add load to the electric system. Funding profiles for the site dictate how and when equipment, processes, and experiments can be operated. Utility programs such as demand charges and demand management plans affect the customer's electrical usage patterns during times of system peaking [3]. Economic factors will not influence the STLF as these factors typically change usage patterns over a longer time range than 24 hours [3]; however, economic factors can be the inspiration for studying a system's load pattern and implementing load

reduction initiatives. As discussed earlier, the STLF is a useful tool for implementing demand management activities.

2.2 Time factors

The three time factors that have the most influence on electrical load are:

1. Seasonal effects
2. Weekly-daily cycle
3. Holidays

Seasonal effects account for the long-term changes in the weather patterns [4]. Hot summer days often create large cooling loads which are more likely to occur during the afternoon and early evening hours. Cold winter nights often create large heating loads which are more likely to occur during the late evening and early morning hours. It is important to note that some facilities such as ORNL use steam for a large percentage of the site's heating needs. Therefore, at ORNL, there will not be as strong a correlation between electric load and air temperature during the winter months as there will be during the summer months. Fig. 2-1 illustrates this seasonality difference by showing the plots of electric load (kW) and air temperature (°C) for a summer week (week of July 10) and a winter week (week of Jan. 8). Seasonal effects are not only weather patterns, but can include popular vacation dates and changes between Daylight Savings Time (DST) [3]. When DST is in effect, the electrical load profile shifts back one hour relative to the profile under Standard Time.

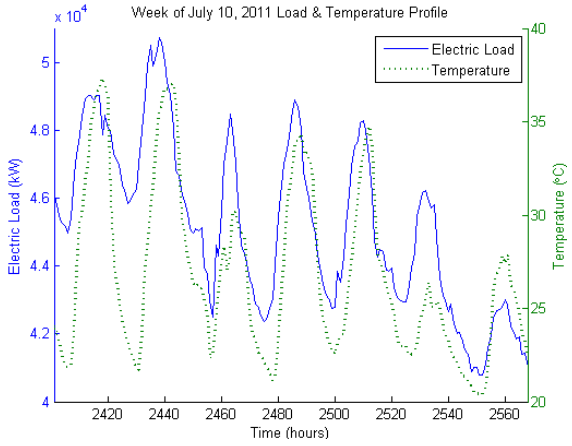
Weekly-daily cycles are electric load patterns that are periodic over the course of a week and during each day [3]-[4]. This periodicity reflects the typical workday for a

facility such as starting, lunch, and quitting time. Fig. 2-1 illustrates this daily and weekly periodicity. The daily load pattern during the summer is shown as a single peak that rises in the morning, peaks in the mid-to-late afternoon, and falls throughout the evening hours. The seventh peak in the summer pattern represents the hourly load profile for a Saturday. It can be seen that although the air temperature is only a few degrees higher than that on Friday, the electrical load peak is lower because Saturday is not a regular business day for ORNL. A similar pattern can be seen in the first peak in the summer pattern. This peak represents the hourly load profile for a Sunday. Even though Sunday is not a regular business day, it is evident the electrical power usage is lower than that of the following Monday even though the temperature peak is higher on Sunday. The difference in magnitude of electric load on Sunday and Saturday is most likely due to the difference in air temperature on both days.

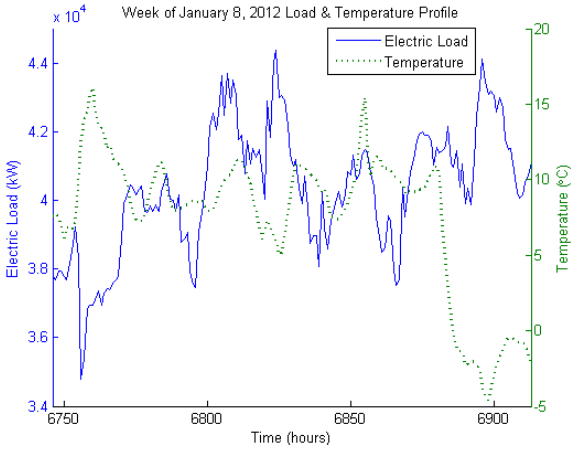
The daily electric load profile for a holiday is similar to that of a weekend profile. The magnitudes of the electric loads are lower. If the holiday is adjacent to a weekend, the load profile can be described as a "long weekend [3]."

2.3 Weather factors

Weather factors have a significant effect on the short-term electric load profile of a power system [1], [3], [5]. Weather-sensitive loads, such as heating, ventilating, and air-conditioning (HVAC) equipment, will have a greater impact on smaller industrial/institutional power systems as these tend to be the larger loads on the system. HVAC equipment cycling on and off can produce electrical load profiles that appear to have random power swings. As the power system load increases, there will be more load



Summer week



Winter week

Figure 2-1. Summer and winter daily electric load and air temperature profile.

diversity, the effect of load cycling will be dampened, and the electric load profile will be smoother.

An illustration of the load's dependence on air temperature is shown in Fig. 2-1 where during the summer week, the electric load peaks occur at approximately the same time as the air temperature peaks and both follow a similar profile. However, during the winter week, the electric load peaks often occur during the lower daily air temperatures. The winter week profile shows that this weather dependence will not always be true. Sometimes, no relation is evident as shown on winter week day three (third air temperature peak). The electrical load profile rises and falls in a similar pattern as the air temperature profile for this day which is the opposite behavior compared to the two previous days.

Other weather factors that can affect the electric hourly load profile are humidity, solar irradiance, wind speed, barometric pressure, and precipitation. High humidity days will make cooling equipment operate for longer duty cycles to remove excess moisture out of the conditioned air. Long durations of high solar irradiance will radiantly heat the interior of buildings forcing the cooling equipment to operate longer and with less diversity. Precipitation has the tendency to reduce the air temperature and thus reduce the cooling load [3]. Wind speed and barometric pressure can also affect the hourly load profile, and often occur in tandem with other factors such as precipitation. Figs. 2-2 and 2-3 illustrate the summer and winter daily profiles of these other weather factors.

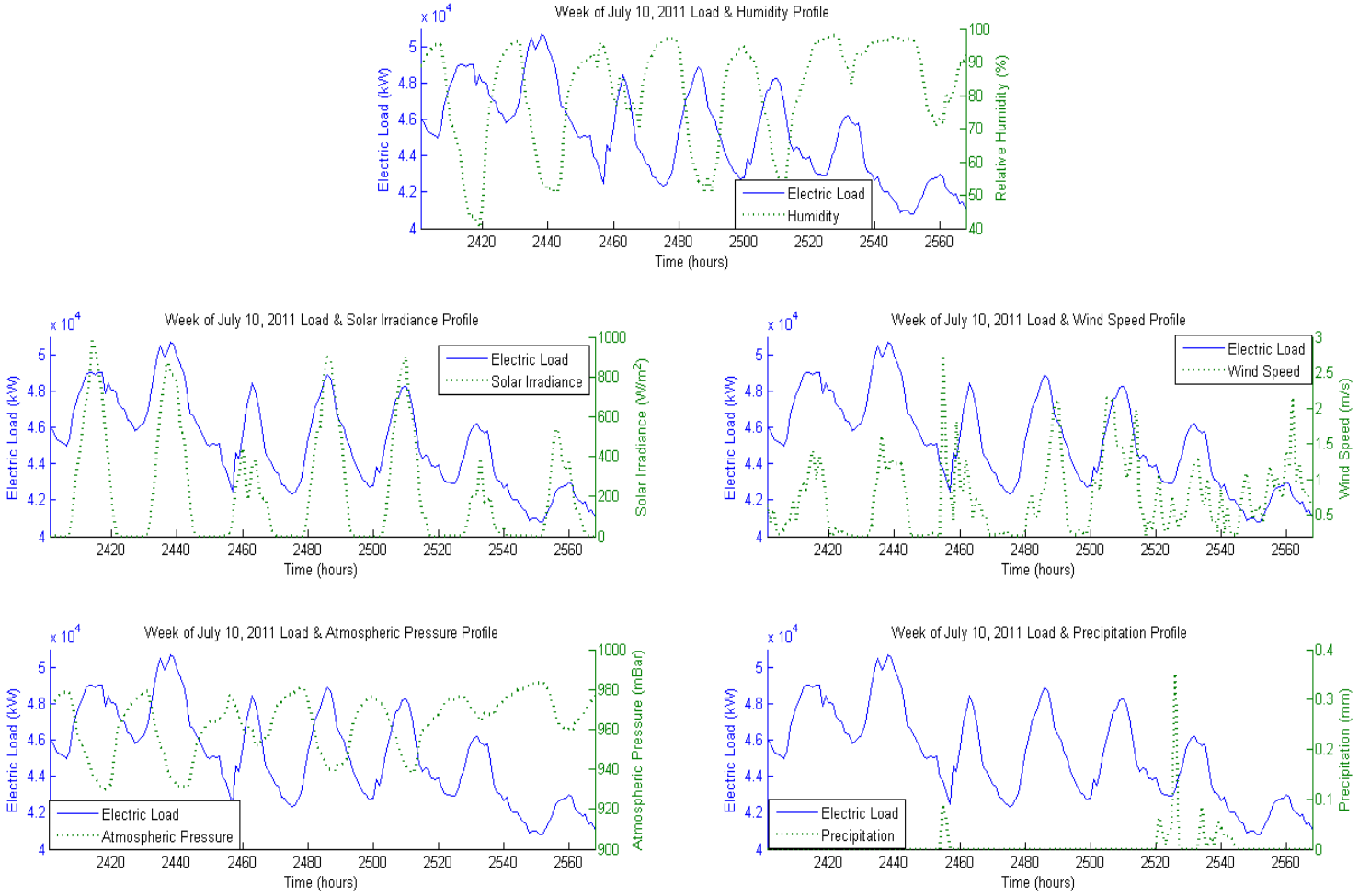


Figure 2-2. Summer daily electric load, humidity, irradiance, wind speed, pressure, and precipitation profile.

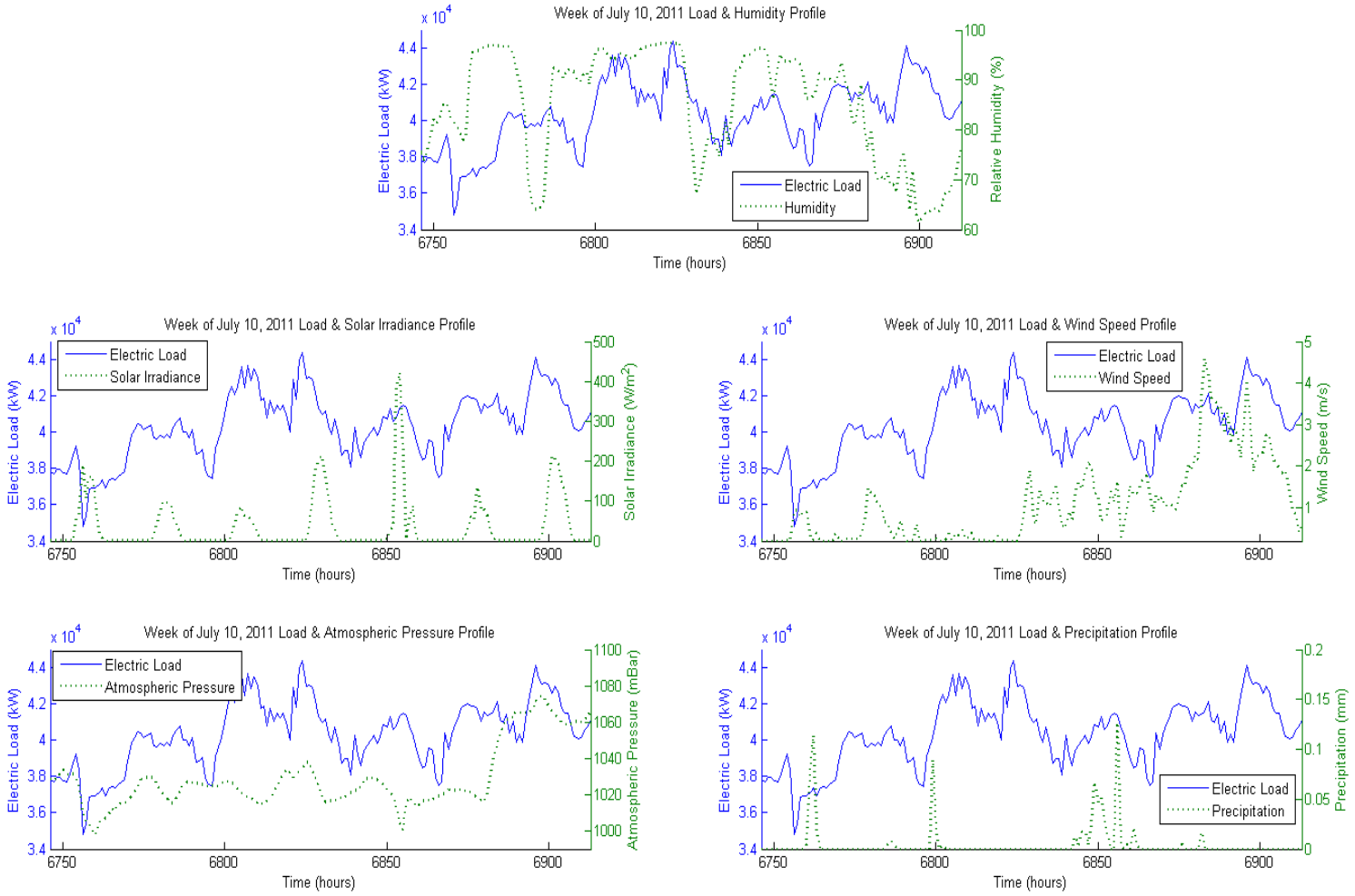


Figure 2-3. Winter daily electric load, humidity, irradiance, wind speed, pressure, and precipitation profile.

2.4 Random Factors

Random factors that influence the electrical load profile consist of all the other random disturbances in the load pattern that cannot be explained by the previous three factors [3]. These disturbances can consist of significant loads that do not have a set operating schedule which makes prediction difficult. Other disturbances such as widespread employee absences (due to sickness, inclement weather, etc.) and planned or unplanned utility system outages can have significant effects on the facility's load profile.

2.5 Load forecasting methods

2.5.1 Multiple linear regression (MLR)

The MLR method calculates the electric load at a specified time *t* using explanatory weather and non-weather variables that are known to have some influence on the electric load [2], [4], [6]- [7]. These variables are identified by correlation analysis with the load [4]. The variables are multiplied by regression coefficients that are found using the least square estimation technique [2], [7].

The MLR electrical load model has the following form:

$$y(t) = a_0 + a_1 x_1(t) + \ldots + a_n x_n(t) + a(t) \qquad (2\text{-}1)$$

where,

$y(t) =$ electrical load.

$x_1(t) \ldots x_n(t) =$ explanatory variables correlated with $y(t)$

$a(t) =$ a random variable with zero mean and constant variance.

$a_0, a_1, \ldots, a_n =$ regression coefficients.

2.5.2 Stochastic time series (STS)

The electric load $y(t)$ is modeled as the output of a linear filter with a random input $a(t)$. This method uses historical load information to forecast the future load [6].

The Autoregressive (AR) process defines the forecasted electric load, $y(t)$, in terms of the previous loads and a random noise signal $a(t)$[2].

$$y(t) = \phi_1 y(t-1) + \phi_2 y(t-2) + \ldots + \phi_p y(t-p) + a(t) \quad (2\text{-}2)$$

The Moving-Average (MA) process defines the forecasted electric load in terms of the current and previous random noise signals. The noise series is constructed from the previous forecast errors [2].

$$y(t) = a(t) - \Theta_1 a(t-1) - \Theta_2 a(t-2) - \ldots - \Theta_{q1} a(t-q) \quad (2\text{-}3)$$

The Autoregressive Moving-Average (ARMA) process defines the forecasted load using a combination of the AR and MA processes [2].

$$y(t) = \phi_1 y(t-1) + \phi_2 y(t-2) + \ldots + \phi_p y(t-p) + a(t) - \Theta_1 a(t-1) - \Theta_2 a(t-2) - \ldots - \Theta_{q1} a(t-q) \quad (2\text{-}4)$$

The model for the ARMA process can be modified to perform forecasting for a time series with a non-stationary mean by a differencing process. This method is known as the Autoregressive Integrated Moving-Average (ARIMA) process [2].

Some time series have periodic behaviors that are due to hourly, daily, weekly, monthly, yearly, or other periodicities. Models for these types of time series are known as seasonal processes. Multiple periodicities, such as hourly and daily cycles, can be modeled with the seasonal process, however, the order of the model increases with each

additional period. The seasonal process model is a modification of the AR, MA, ARMA, or ARIMA process [2].

The Transfer Function (TF) model utilizes one of the previously discussed models to represent historical load and a white noise term with one or more other variables that affect the load such as temperature or humidity. The effects of these other variables can be modeled using a transfer function as shown in Fig. 2-4.

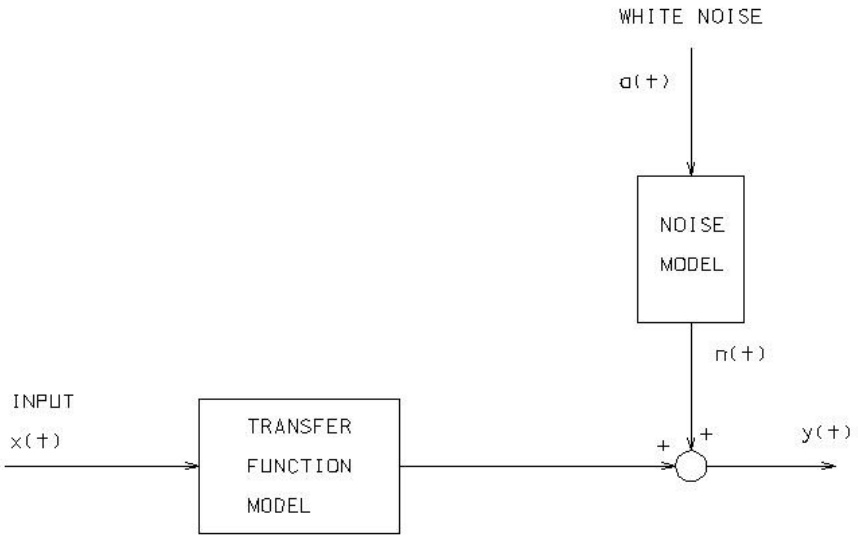


Figure 2-4. Transfer Function (TF) model [2].

The input series $x(t)$ represents one of these additional variables. The series $n(t)$ is defined as a colored noise that is modeled as a white noise and previous $\mathrm{n}(t)$ values [2].

2.5.3 General Exponential Smoothing (GES)

The General Exponential Smoothing (GES) load forecasting method models the observed load as [2], [8]:

$$y(t) = \beta(t)^T f(t) + \varepsilon(t) \tag{2-5}$$

where,

$f(t) =$ vector of linearly independent and stationary fitting functions

$\beta(t) =$ coefficient vector that is locally constant

$\varepsilon(t) =$ white noise

T = transpose operator

The fitting functions are selected so that the forecasted load for the next 24 hours is a linear combination of these functions [8]. Load forecasts can be calculated using the known fitting functions and estimates of future $\beta(t)$ using [2]:

$$y(N + l) = f^T(l)\beta(N) \tag{2-6}$$

where,

N = last hour the load was known

l = lead time in hours

The estimated coefficient vector $\beta(N)$ is found using a weighted least squares criterion to minimize the function [2], [8]:

$$\sum_{j=0}^{N-1} w^j [y(N - j) - f^T(-j)\beta]^2 \quad 0 < w < 1 \tag{2-7}$$

thus,

$$\beta(N) = F^{-1}(N)h(N) \tag{2-8}$$

where,

$$F(N) = \sum_{j=0}^{N-1} w^j f(-j)f^T(-j) \tag{2-9}$$

$$h(N) = \sum_{j=0}^{N-1} w^j f(-j)y(N - j) \tag{2-10}$$

Reference [8] describes (2-7) as "the exponentially weighted sum of the squared deviations of the load model from the observed data." The rate at which past errors are dismissed is controlled by the constant w.

Updates to the coefficient vector and forecast are made using [2], [8]:

$$\beta(N+1) = L^T\beta(N) + F^{-1}f(0)[y(N+1) - y(N)] \tag{2-11}$$

$$y(N+1+l) = f^T(l)\beta(N+1) \tag{2-12}$$

where,

$$F = \lim_{N\to\infty} F(N) \tag{2-13}$$

and L is the transition matrix that satisfies,

$$f(t) = Lf(t-1) \tag{2-14}$$

2.5.4 State Space (SS)

ARMA load forecasting models can be converted to State Space (SS) models and vice versa. One difference between the two methods is the SS models often provide more structure than some ARMA models allowing a more concise presentation and manipulation. The State Space load forecasting method has many variations, but they all model the load as a state variable using a state space formulation. Often, this state formulation consists of two equations [2]:

State Space Equations:

$$X(k+1) = \Phi(k)X(k) + W(k) \tag{2-15}$$

Measurement Equation:

$$Z(k) = H(k)X(k) + V(k) \tag{2-16}$$

where,

$X(k) = (n \text{ x } 1)$ process state vector at time t_k

$\Phi(k) = (n \text{ x } n)$ state transition matrix relating $X(k)$ to $X(k+1)$ when no forcing function exists

$W(k) = (n \text{ x } 1)$ white noise with a known covariance $Q(k)$

$Z(k) = (m \text{ x } 1)$ vector of load measurements at time t_k

$H(k) = (m \text{ x } n)$ matrix relating $X(k)$ to $Z(k)$ without noise

$V(k) = (m \text{ x } 1)$ load measurement error which is a white noise with a known covariance $R(k)$

SS models are not as common as ARMA models for load forecasting. One possible reason for fewer SS models is because the ARMA models require fewer explanatory variables and parameters [3], and the noise covariance matrices $Q(k)$ and $R(k)$ are difficult to estimate [2].

2.5.5 Knowledge-Based Expert Systems (KBES)

Knowledge-Based Expert Systems (KBES) attempt to emulate the decision-making abilities of power system operators when forecasting future system load. These decisions are converted to formal logical steps that can be programmed. The systems are constructed from the expert knowledge of the system operator(s), historical electrical load, and relevant weather data [9]. The KBES is a computer program, but it is not described as algorithmic. Reference [2] states the KBES is a program that "can reason, explain, and have its knowledge base expanded as new information becomes available to it."

The KBES programmer extracts the intuitive relationships between load and weather, time, date, and season from the system operator, or expert, to construct the knowledge-base. The KBES knowledge extraction process is known as the acquisition module [2]. The knowledge-base consists of rules followed by IF-THEN rules and mathematical expressions that are used to make the forecasts. The knowledge extraction process is performed off-line; however, new information can be added to the knowledge base allowing the system to be continually updatable [9].

The reasoning module of the KBES is known as the inference engine [2]. It documents the steps performed by the program at arriving at a forecast.

2.5.6 Artificial Neural Network (ANN)

An Artificial Neural Network can be described as a mathematical tool that mimics the thought processes of the human brain. ANNs were first applied to load forecasting in the late 1980's [10]-[11]. They are described as a multivariate, nonlinear, and nonparametric method which makes the good candidates for modeling complex nonlinear systems [10], [12]-[13]. ANNs have good performance in data classification and function fitting [14].

Neurons are the basic processing components of ANNs. The neurons are programmed to behave similarly to the neurons in the brain by receiving inputs, processing those inputs, and producing an output. The neuron is shown schematically in Fig. 2-5 [10].

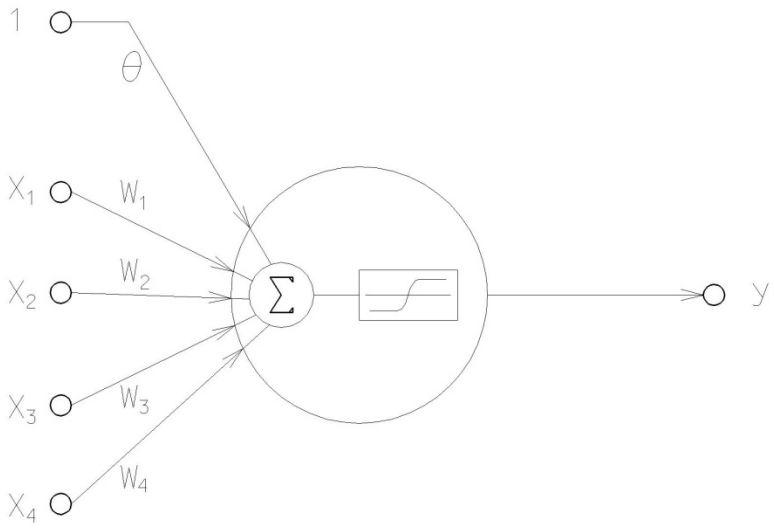


Figure 2-5. An artificial neuron model [10].

The neuron model consists of the linear combination of one or multiple numerical inputs (represented by X_i in Fig. 2-5) and a constant input term (represented as an input of 1 in Fig. 2-5). Each variable input X_i is adjusted by a unique weight, w_i, and the constant input of 1 is adjusted by a variable bias, Θ. The linear combination of these inputs and bias are input to a nonlinear activation function whose output is the output of the neuron. The training of ANNs requires the activation function to be nondecreasing and differentiable [6], [10]. The most common activation functions used in ANNs are the linear function $y = x$, or some variation of the bounded sigmoid function such as the logistic function $y = 1/(1 + e^{-x})$ [6], [10], [15].

The architecture of ANNs can vary, but the most commonly used is the multilayer perceptron (MLP) as shown in Fig. 2-6 [10]-[11], [16].

Fig. 2-6 shows an ANN that consists of two layers, an output layer (to which the output nodes are connected), and a hidden layer (located between the inputs nodes and

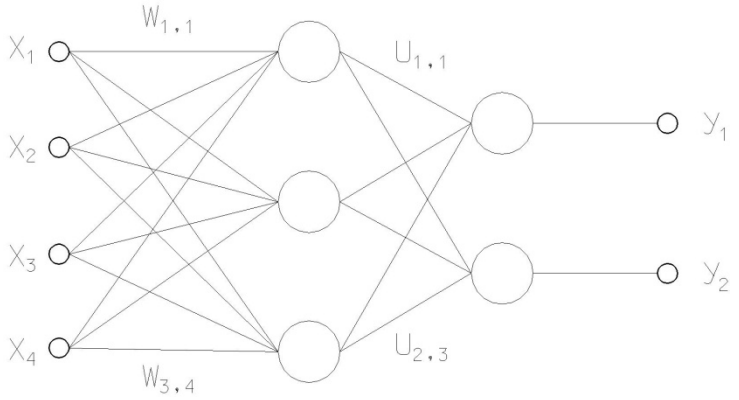


Figure 2-6. Two-layer, feed-forward, neural network [10].

output layer). The neurons in a layer can share inputs, but they are not connected to one another. If the ANN is a feed-forward network, then the outputs of one layer are connected as the inputs to the next layer [17], [26]. Often, one hidden layer is adequate to approximate any continuous function [10]. Depending on the network topology, the ANN can have multiple outputs.

Fig. 2-6 can be represented mathematically as follows if the hidden layer neurons' activation functions are logistic, and the output layer neurons' activation functions are linear [10]:

$$y_k = \sum_{j=1}^{3}\left(u_{kj} \times \frac{1}{1+exp\left(-\sum_{i=1}^{4} w_{ji}x_i+\Theta_j\right)}\right) + \Theta_k \qquad (2\text{-}17)$$

The weights and biases of the ANN are determined through a training algorithm that minimizes a loss function. The backpropagation learning algorithm is often cited in the literature as being preferred [6], [10], [12], [15], [17]-[22]. This learning algorithm is based on the Widrow-Hoff error correction rule where the difference between an output and a target value forms an error signal. The weights and biases of the ANN are adjusted

to minimize the error signal between its output and target value [6], [12]. Since ANNs can have more than one hidden layer, determining the weights and biases for the hidden layers can be complex. The generalized delta rule can be used where the error signal at the output is propagated back to the hidden layers until it reaches the input layer [6], [18], [20], [26]. This error minimization continues until some stop criteria is reached, such as error tolerance or number of iterations. The training sequence allows the ANN to learn the relationship between its inputs and outputs and store these learned parameters to allow future operations without retraining.

An ANN for load forecasting can be trained on a training set of data that consists of time-lagged load data and other non-load parameters such as weather data, time of day, day of week, month, and actual load data [5]. Some ANNs are only trained against days with data similar to the forecast day [21]-[22]. Once the network has been trained, it is tested by presenting it with predictor data inputs. The predictor data can be time-lagged load data and forecasted weather data (for the next 24 hours) [5]. The forecasted load output from the ANN is compared to the actual load to determine the forecast error.

Forecast error is sometimes presented in terms of the root mean square error (RMSE) [5], [10], [12], [14], as shown in (2-18) with units of kW, but more commonly in terms of the mean absolute percent error (MAPE) [1], [4]-[5], [10]-[14], [16], [18]-[19], [21]-[22] as shown in (2-19) with units of percent.

$$RMSE = \sqrt{\frac{1}{N} \times \sum_{t=1}^{N} (y_t - \hat{y}_t)^2} \tag{2-18}$$

$$MAPE = \frac{1}{N} \times \sum_{t=1}^{N} \left(\frac{|y_t - \hat{y}_t|}{y_t} \times 100 \right) \tag{2-19}$$

where,

$N =$ number of samples (24 for 24-hour load forecast)

$y_t =$ actual load at time t

$\widehat{y_t} =$ forecasted load at time t

It should be noted that an ANN trained on a specific power system's load and weather data will be system dependent. The ANN generated for that system will most likely not perform satisfactorily on another power system with differing characteristics. It is possible the same ANN architecture may be reused on the new system, but retraining will be required.

An ANN trained using the backpropagation technique can succumb to overfitting. Reference [10] defines overfitting of an ANN as "estimating a model that fits the data so well that it ends by including some of the error randomness in its structure, and then produces poor forecasts." The overfitted ANN can replicate in-sample (training) data with very low errors, but when presented with out-of-sample data, the results have high errors. This type of ANN is not capable of generalizing the input-output data relationships. Overfitting can occur due to overtraining and excessive network complexity [5], [10].

Overtraining can be avoided by using cross-validation. This process splits the original training data into a training set and validation set [10], [14]-[15]. The ANN parameters are trained on the training set and tested every few iterations on the validation set. When the performance of the validation set starts to decrease, training is ended [10].

As previously stated, overly complex ANNs can overfit its training data. The ANN attempts to track down each data point in the training set [10]. Keeping the model simple will help produce smooth models that usually forecast better than complex models. ANNs that are designed with complex arrangements and classes on input data and high numbers of hidden neurons must estimate a very large number of weights and biases during training. An ANN trained under these conditions with a small input data set will have estimated many internal weights and biases on a comparatively small sample set of data. This will lead to a lack of generalization of the ANN's performance on out-of-sample data and result in high errors [10], [14]. A method, such as correlation analysis, should be applied to the input data in an effort to reduce the amount of data to only what is necessary [5], [15]-[16].

This chapter presented a summary of the existing literature for 24-hour load forecasting methods. Table 2-1 summarizes the presented load forecasting techniques. The latter part of the chapter focused on ANN load forecasting methods that are used in Chapter 3 to create a 24-hour load forecasting ANN.

Table 2-1
Short-term load forecasting methods.

Method	Load Model	Popularity for STLF
Multiple linear regression (MLR)	Linear equation with regression coefficients and a random variable	High
Stochastic time series (STS) Various models: Autoregressive (AR) Moving-Average (MA) Autoregressive Moving-Average (ARMA) Autoregressive Integrated Moving-Average (ARIMA) Transfer Function (TF)	Linear equation expressed in terms of previous load values (time sereies) modeled as a linear filter with a white noise input	Highest
General Exponential Smoothing (GES)	Linear combination of known functions of time and a noise component	Very low
State Space (SS)	General method that uses state space and measurement equations. It can incorporate MLR, STS, or GES methods.	Very low
Knowledge-Based Expert Systems (KBES)	Computer program consisting of IF-THEN statements and mathematical expressions. Relies on an existing "knowledge-base"	Low
Artificial Neural Network (ANN)	A multivariate, nonlinear, and nonparametric computer program. Must be trained on historical data.	High

Chapter 3
Materials and Methods

This chapter describes the material and methods used to create ANNs that perform 24-hour load forecasts.

MATLAB® (version R2012b) by Mathworks®, Inc., was the computer software used to create and implement the 24-hour load forecast for ORNL. The Neural Network Toolbox in MATLAB® provides built-in functions and applications to assist in modeling nonlinear systems. It supports ANN training, validation, testing, and simulation with hardcode and graphical user interface (GUI) applications [23]. The MATLAB® code for this research was executed on a 2.10 GHz Intel® Core™2 Duo CPU running Microsoft Corp. Windows® 7 Home Premium (32-bit) Service Pack 1. Appendix A contains the 24-hour load forecast MATLAB® m-file code.

3.1 Input Data

One year (spanning April 1, 2011 – March 31, 2012) of electrical load (File 1, ORNL_2011_2012.xlsx) and weather data (File 2, Weather_Data.xlsx) was recorded for ORNL and logged in two Microsoft Excel® spreadsheets. It should be noted that 2012 was a Leap Year, so there are 366 days of data collected. This data was used for training and testing of the 24-hour load forecasting ANN. The load data was collected from a database containing data recorded from power meters located at the ORNL 161 kV substation. The weather data was collected from a database maintained by the National Renewable Energy Laboratory (NREL) containing data recorded from a rotating

shadowband radiometer weather station located at ORNL. The weather data is accessible on the internet at the following website: http://www.nrel.gov/midc/ornl_rsr/.

The load data spreadsheet contains the electrical load for ORNL recorded for the previous half-hour, however, the 24-hour forecast ANN only requires hourly metered load data as inputs, so the half-hour values were not used. The remaining original columns of the load spreadsheet were neglected as they were not required for the ANN. Additional columns were added to the original load data spreadsheet to indicate day-of-week and serial date as these values were also required as inputs to the ANN. Serial date is a Microsoft Excel® numerical representation of the calendar date as a total number of days starting with January 1, 1900 as serial date 1. For example, April 1, 2011 is equivalent to serial date 40634.

The weather data spreadsheet contains the average weather data for ORNL recorded for the previous hour. The spreadsheet contains values for the following parameters:

1. Average global horizontal irradiance (W/m^2) – the sum of direct normal irradiance, diffuse horizontal irradiance, and ground-reflected solar radiation.
2. Average dry-bulb air temperature (°C) – the air temperature measured by a thermometer.
3. Average relative humidity (%) – the amount of water vapor in the air expressed as the ratio between the measured amount and the maximum possible amount (saturation point at which water condenses as dew).

4. Average wind speed (m/s) – wind speed measured by a 3-cup anemometer mounted approximately 42 feet above ground level.
5. Average estimated atmospheric pressure (mBar) – estimated atmospheric pressure based on the weather station's elevation and air temperature.
6. Average precipitation (mm) – amount of precipitation as measured by a tipping bucket precipitation gauge mounted approximately 38.5 feet above ground level.

The load data was time and date-stamped in U.S. Eastern Daylight Savings Time. Therefore, for the hour spanning 1-2 am on Nov. 6, 2011, there are four half-hour load data records as a result of the extra hour due to Daylight Savings Time. Also, for the hour spanning 2-3 am on March 11, 2012, there are no load data records as a result of the lost hour due to Daylight Savings Time.

The weather data was time and date-stamped in U.S. Eastern Standard Time. To account for the time difference between the weather data's Standard Time and load data's Daylight Savings Time time-stamps, the first weather record was chosen as hour "0" which represents average weather data for the 11:01 pm March 31, 2011 - 12 am April 1, 2011 hour. Doing this aligns the weather data with the load data's first record which is the average load data for 1 am DST, April 1, 2011 (12 am EST April 1, 2011). Therefore, during Daylight Saving Time, the time stamps of the load and weather records are off by one hour, but the data values were recorded at the same time. During Standard Time, the time stamps of the load and weather records match.

The weather data spreadsheet's Hour records were imported into the ANN to document the time of day for each input data (load and weather). The weather data's Hour records were used because this data was recorded in EST, so there will not be extra or missing Hour values during the change to and from DST as there would be if the load data Hour records were utilized.

The load data spreadsheet has a Day of Week column which contains a numerical representation for the day of the week each load data value was recorded. Table 3-1 defines these numerical values for Day of Week.

3.2 Preprocessing

3.2.1 Date

The metered/recorded load and weather data parameters were imported into MATLAB® as individual column vectors. An additional conversion was required on the serial date vector imported from the load data Excel® spreadsheet because the MATLAB® serial date system starts on January 0, 0000 which is different from the starting date in Excel® which is January 1, 1900.

Table 3-1
Day of week numerical value.

Day of Week	Numerical Value
Sunday	1
Monday	2
Tuesday	3
Wednesday	4
Thursday	5
Friday	6
Saturday	7

3.2.2 Time Lags

Chapter 2 noted that future electrical load is greatly influenced by previous load values. As a result, it was imperative that previous load values were used as inputs to the ANN. Additional column vectors were created from the original input data to store time-lagged load data. If d represents the forecast day, and t represents the current forecasted hour, then the time-lagged input load data is as shown in Table 3-2.

The amount of time lag was determined from correlation analysis which is discussed later in this chapter.

Table 3-2
Time-lagged input load data.

Forecast-Day Lag	1-Day Lag	2-Day Lag	1-Week Lag
n/a	kW(d-1, t)	kW(d-2, t)	kW(d-7, t)
kW(d, t-1)	kW(d-1, t-1)	kW(d-2, t-1)	kW(d-7, t-1)
kW(d, t-2)	kW(d-1, t-2)	kW(d-2, t-2)	kW(d-7, t-2)
kW(d, t-3)	kW(d-1, t-3)	kW(d-2, t-3)	kW(d-7, t-3)
kW(d, t-4)	kW(d-1, t-4)	kW(d-2, t-4)	kW(d-7, t-4)

3.2.3 Calendar Designations

The date-stamps for each input load data were further processed to create month, weekend, and holiday records.

The month record contained a numerical representation of the month (1 through 12) each load data value was recorded.

The weekend record contained a binary value (0 or 1) for each input data which indicated the recorded value occurred on a weekend (binary value 1) or not on a weekend

(binary value 0). Weekends were defined as dates whose Day of Week equaled 1 (Sunday) or 7 (Saturday).

The holiday record contained a binary value (0 or 1) for each input data which indicated the recorded value occurred on a holiday (binary value 1) or not on a holiday (binary value 0). Holidays were defined as dates designated by ORNL's operations calendar as official company holidays (Appendix B).

3.3 Correlation Analysis

A list of all input data variables available for use in the ANN is shown in Table 3-3. Correlation analysis was performed on the input data to optimize the ANN's training and load forecasting algorithms. This analysis identified which input data is correlated to the actual load data and which data was neglected. This analysis also aided the selection of the number of time lags required for the load data.

Fig. 3-1 shows the one year electric load profile for ORNL. A detailed study of the load profile reveals obvious trends to the load values. The load profile displays a seasonal behavior. The higher load values in the first third of the dataset occurred during the summer season (May-August). The load values starting at 1 to approximately 1000 hours occurred during the 2011 spring season (April). The load values starting at approximately 3500 hours to approximately 8700 hours occur during 2011 fall (September-October), 2011/2012 winter (November-February), and 2012 spring (March) seasons. The term season is used here to represent different time periods with respect to the load profile trends, not necessarily the calendar seasons which have defined starting

and ending dates. This seasonality implies a correlation between load and month and the load profile within those months associated with a season will have similar patterns.

Table 3-3
Complete list of input data variables.

Electric Load	Weather	Date/Time
kW(d, t-1)	Irradiance	Serial Date
kW(d, t-2)	Air Temperature	Month
kW(d, t-3)	Humidity	Day of Week
kW(d, t-4)	Wind Speed	Hour of Day
kW(d-1, t)	Air Pressure	Weekend
kW(d-1, t-1)	Precipitation	Holiday
kW(d-1, t-2)		
kW(d-1, t-3)		
kW(d-1, t-4)		
kW(d-2, t)		
kW(d-2, t-1)		
kW(d-2, t-2)		
kW(d-2, t-3)		
kW(d-2, t-4)		
kW(d-7, t)		
kW(d-7, t-1)		
kW(d-7, t-2)		
kW(d-7, t-3)		
kW(d-7, t-4)		

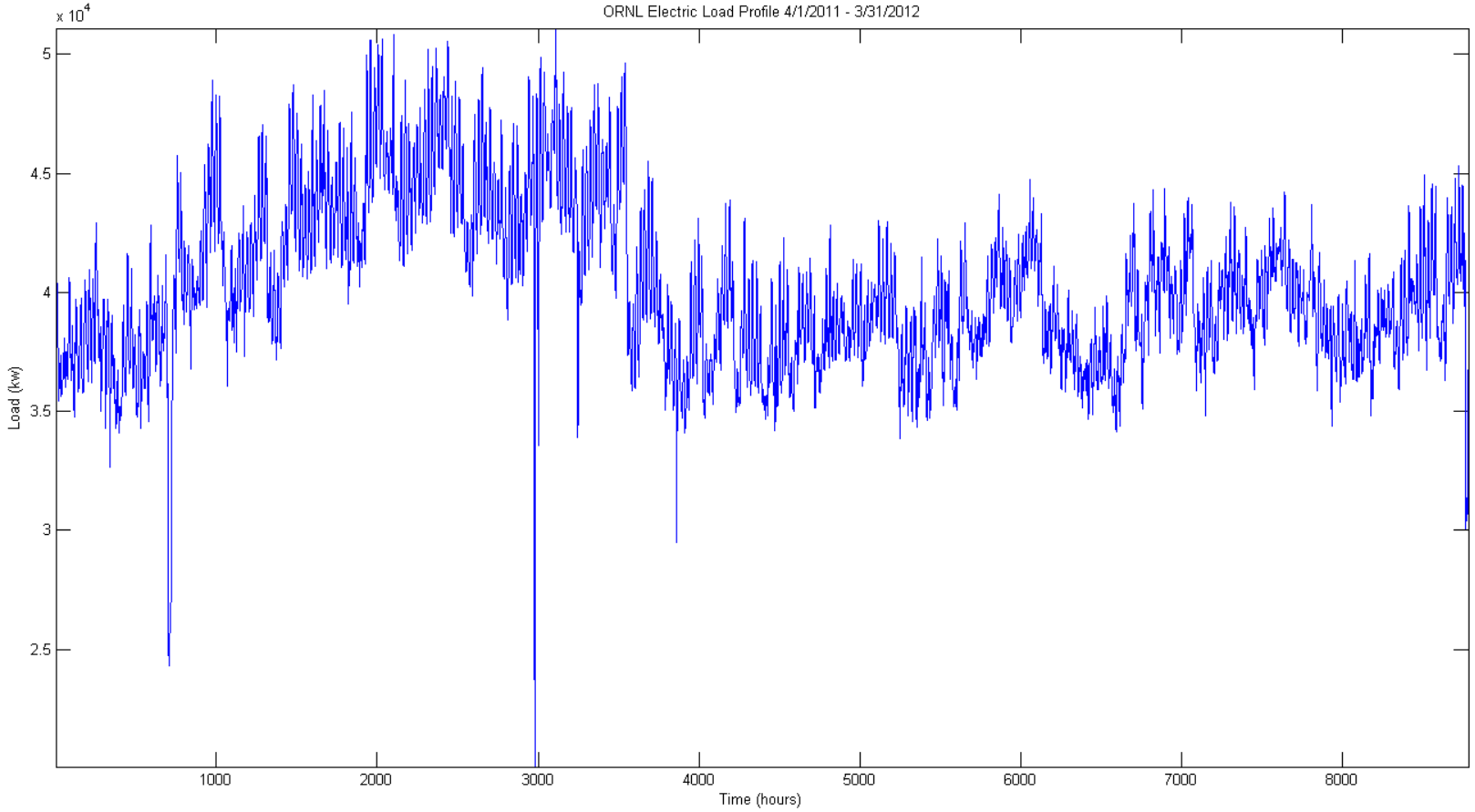


Figure 3-1. ORNL one year electric load profile.

Fig. 3-1 also reveals the significance a holiday has on the load profile. As an example, the dip in the load profile between approximately 6000 and 7000 hours is the load data for the week before, during, and after Christmas. The Monday following New Year's Day also has a reduced load level, but the load profile returns to a more seasonal pattern the following Tuesday. Other holidays have a similar reduction in load on the day of the holiday, but none of them have as dramatic an effect as the Christmas holiday.

Fig. 3-2 shows the 24-hour load and temperature profiles for a Wednesday from each season (summer, fall, winter, and spring).

The summer season plot in Fig. 3-2 shows the high correlation between load and temperature. The winter season plot shows a high negative correlation between load and

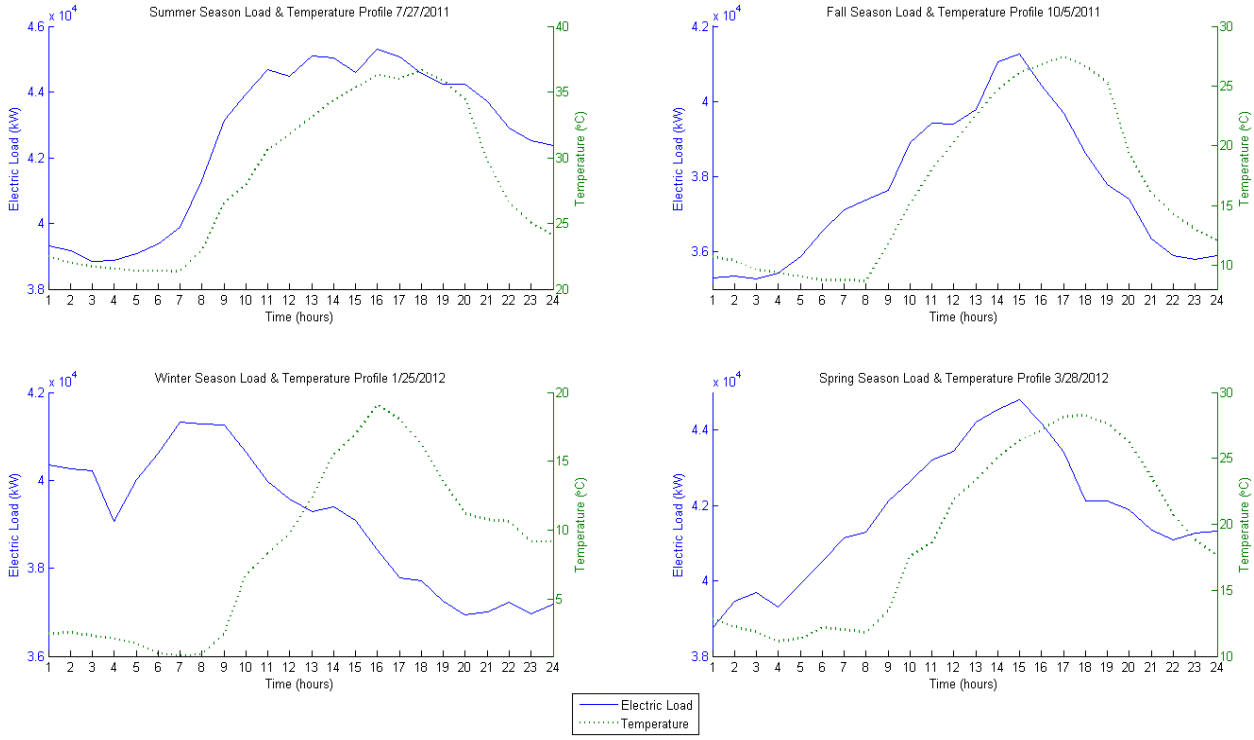


Figure 3-2. 24-hour electric load profiles showing seasonality.

temperature. An explanation for the profiles of these two plots is the fact that cooling demand is high in the summer and heating demand is high in the winter. During the hot afternoons in the summer, ORNL's cooling systems will be operating causing high electricity demand. Also, during the cold nights and early mornings during the winter, ORNL's heating systems will be operating causing high electricity demand. The fall and spring season plots have similar profiles. They show a positive correlation between load and temperature, but the temperature profile has a slight offset, so the correlation for fall and spring seasons will often be less than that for the summer season.

Correlation between load and temperature is evident in the scatter plot of Fig. 3-3.

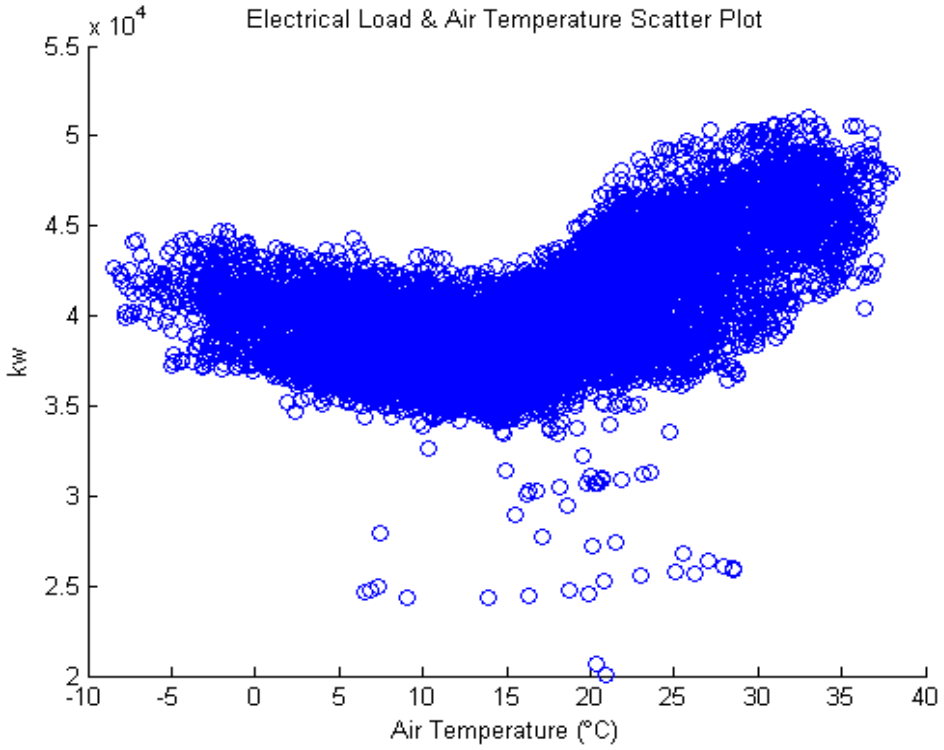


Figure 3-3. One year correlation between electric load and air temperature.

Fig. 3-3 shows that during the colder days, the load is high, but as the air temperature increases, the load decreases. At approximately 15 °C, the load stops decreasing and begins to increase with increasing temperature. The load values have overall higher maximums at the upper bounds of air temperature than at the lower bounds. Scatter plots of load and the other weather variables are shown in Chapter 4.

The perceived correlations shown in the scatter plots can be quantified using Pearson's correlation coefficient. This coefficient gives an indication of the linear relationship between two variables [24]. The coefficient can range between -1 and +1 where -1 indicates a high negative correlation, 0 indicates no correlation, and +1 indicates a high positive correlation. As stated, Pearson's correlation coefficient is a measure of linear correlation; however, the scatter plots in Fig. 3-3 and Fig. 3-4 show that often, the relationship between the load and the weather variables is nonlinear. Pearson's correlation coefficient can be applied to the weather and load datasets for load forecasting if the dataset range is small enough to present a linear relationship. If the air temperature and load datasets are restricted to a set number of weeks during the colder season (winter), then there appears to be a negative linear relationship between air temperature and load. If the air temperature and load datasets are restricted to a set number of weeks during the hotter season (summer), then there appears to be a positive linear relationship between air temperature and load.

It is advantageous to maintain a smaller input dataset to the ANN so that a linear correlation analysis can be performed, and from this analysis, only the relevant input data will be presented to the ANN. This correlation analysis was performed on all input data

shown in Table 3-3 for the training and forecast predictor datasets. The minimum correlation coefficient value is defined in the simulation code. This value can be increased or decreased to allow less or more input data types, based on their correlation coefficients, to be presented to the ANN.

3.4 Artificial Neural Network

3.4.1 Training and Minimizing the Forecast

After the correlation analysis was performed on the training and forecast predictor datasets, the number of hidden layers, or neurons, in the ANN was defined and the ANN was created for the user-defined forecast day. The built-in Matlab® Levenberg-Marquardt optimization training function was used to perform the backpropagation training of the feed-forward ANN [23]. This process iteratively updated the internal weight and bias values of the ANN to obtain a low error output when utilizing the training predictor dataset and a target dataset. The target dataset consists of the actual load values for a given predictor dataset.

After training, the ANN was tested using only the training predictor dataset. This step allows the user to verify the trained ANN can produce low error forecasts on in-sample data.

After testing, the ANN performed a 24-hour forecast using only the forecast predictor dataset. The results of this forecast were stored, and the entire ANN training, testing, and forecasting process was repeated a set number of times with the intention of reducing the forecast error.

Two methods were concurrently used to minimize the forecast error. Method 1 assumes the 24-hour forecast's MAPE will be minimized if the trained ANN's testing step MAPE was minimized. After each training iteration, if the ANN's training MAPE was reduced, the newly trained ANN would replace the previously stored ANN. Method 2 assumes the 24-hour forecast's MAPE will be minimized if the MAPE of the previous day's 24-hour forecast is minimized. This requires a STLF to be performed for the day prior to the forecast day. Since the previous day's load is known, an accurate MAPE can be calculated. The prior day's MAPE was iteratively minimized in a similar manner as Method 1 above. When a new minimum MAPE for the previous day was stored, the trained ANN associated with that minimum MAPE was stored.

Once the training error was minimized, the 24-hour forecast associated with Method 1 was stored. Method 2 created a new 24-hour forecast for the forecast day using the stored ANN obtained in Method 2.

This entire error minimization and forecast storage process would repeat a fixed number of iterations. At the end of each iteration, the stored 24-hour forecasts were added to the previously stored 24-hour forecast so that at the completion of the forecasting algorithm, the 24-hour forecast values were averaged. This process occurred for both error minimization methods resulting in two separate (Method 1 and Method 2) 24-hour load forecasts for the same forecast day. The intent of the STLF is to forecast the next day's 24-hour load; the actual load values would not be known. Therefore, it was assumed the day prior to the forecast day had a similar load profile as the forecast day, so its load values would be used to calculate an approximate MAPE for both

minimization methods' forecasts. The method that had the lower approximated MAPE was defined as the final 24-hour load forecast.

The reasoning behind the error minimization and repeating forecasting steps is that the Matlab® Neural Network Toolbox assumes random initial values when it begins the ANN training algorithm. Sometimes the initial values lead to an ANN that outputs very low error; other times, they lead to an ANN with low error, but not minimum error. Averaging the multiple runs of the 24-hour forecasted loads created a more consistent forecast MAPE. The process flow diagram of Fig. 3-4 illustrates the main steps in the Matlab® ANN program.

The absolute percent errors for each hour forecast and mean absolute percent error for each 24-hour forecast were calculated.

3.4.2 Previous Load Updates

If lagged-load data was used in the forecasted predictor dataset for the ANN, then the applicable lagged-load data values in the forecasted predictor dataset were updated after each hour's load was forecasted as in [25]. For example, if the previous one-hour ago load, kW(d, $t-1$), was used in the forecast predictor, the forecasted load for the current hour, t, would be inserted into the forecasted predictor dataset's "previous one-hour ago load" for the next hour, $t+1$. This ensures that each forecasted load is used as an input to future forecasts.

This chapter described the material and methods used to create ANNs that perform 24-hour load forecasts. The following chapter presents the results of 24-hour

forecasts for various days. Chapter 4 also explores the bounds on the system load and weather specific to ORNL and the effects these have on the forecasted load.

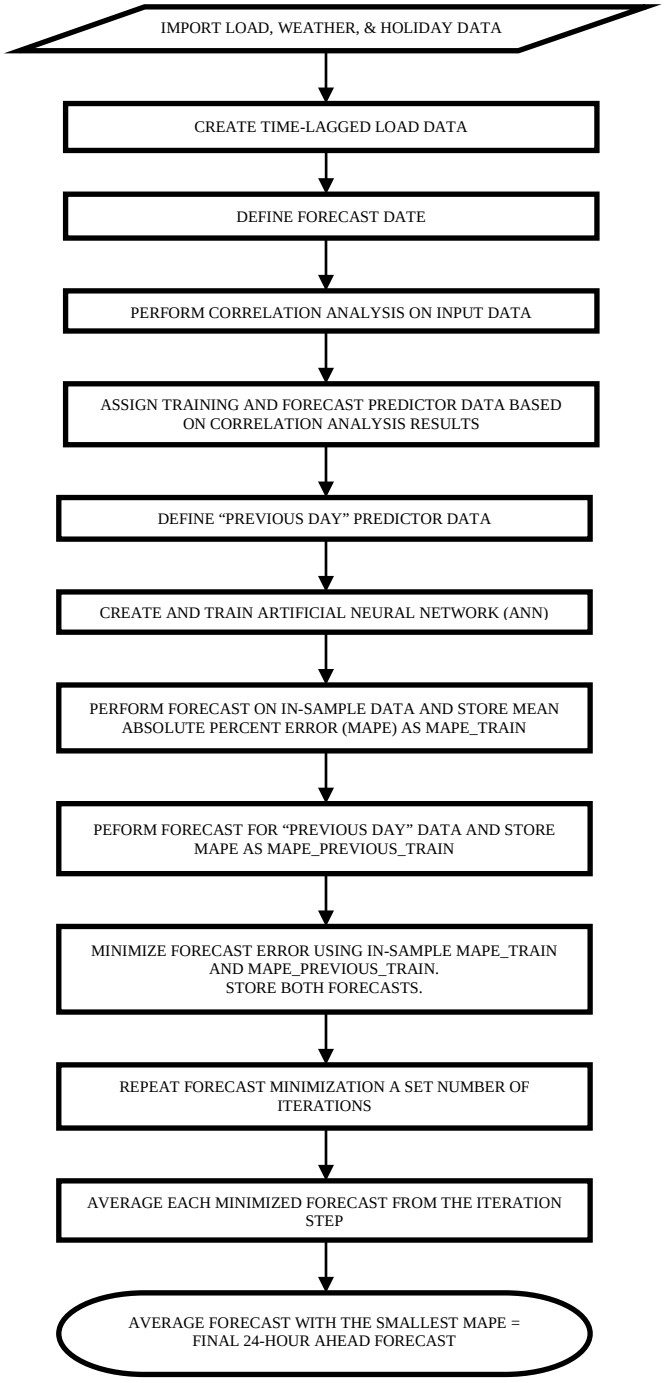


Figure 3-4. Matlab® ANN program process flow diagram.

Chapter 4

Results and Discussion

This chapter presents the results of various 24-hour load forecasts using trained ANNs. Scatter plots for the predictor inputs are presented to visually see the correlations and trends between the predictors and the load. Correlation analysis results for selected 24-hour load forecast days are tabularized and discussed to show how this step reduces the size and complexity of the predictor matrices. Load forecast results one week out of each season (spring, summer, fall, and winter) are presented along with 24-hour load forecast profile plots for selected days.

4.1 Predictor Scatter Plots

As noted in Chapter 3, the relationship between the load and other variables can be visualized via a scatter plot of the associated datasets. Figs. 4-1, 4-2, 4-3, and 4-4 are scatter plots showing the relationship between load and time-lagged load at various time lags. Fig. 4-1 is the scatter plot of load and the loads of the previous four hours.

Fig. 4-1 shows a strong positive correlation between the forecast-time load and the previous four hour load values. The grouping of data points are very linear in the $(d, t-1)$ lag. This is expected as there usually is not a large change in load value in one hour. As the time-lag increases, the correlation begins to decrease as evident in the growing area in which the data points are scattered.

Fig. 4-2 is the scatter plot of load and the previous day's load at the forecast time and the previous four hours.

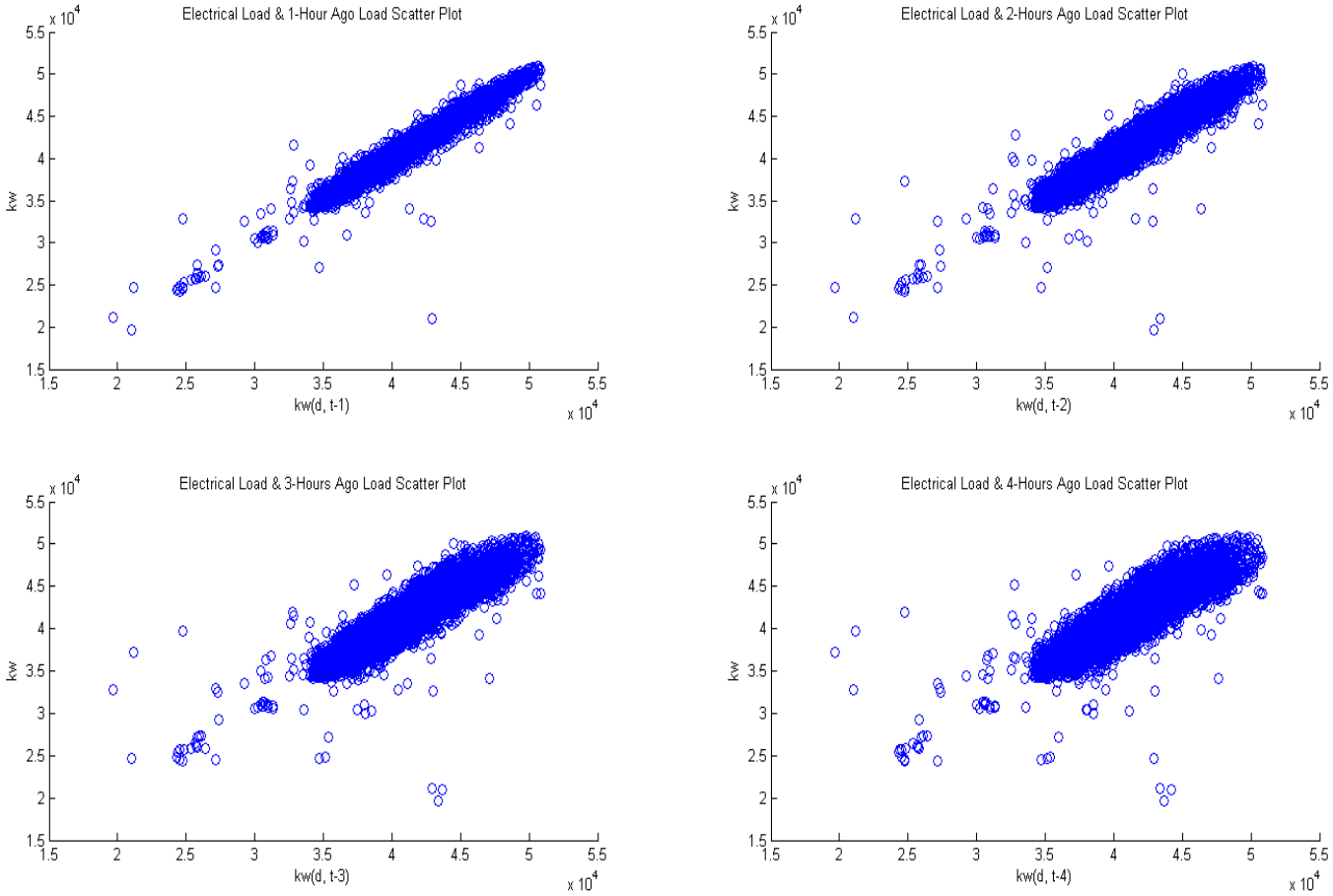


Figure 4-1. One year correlation between electric load and same-day time-lagged loads.

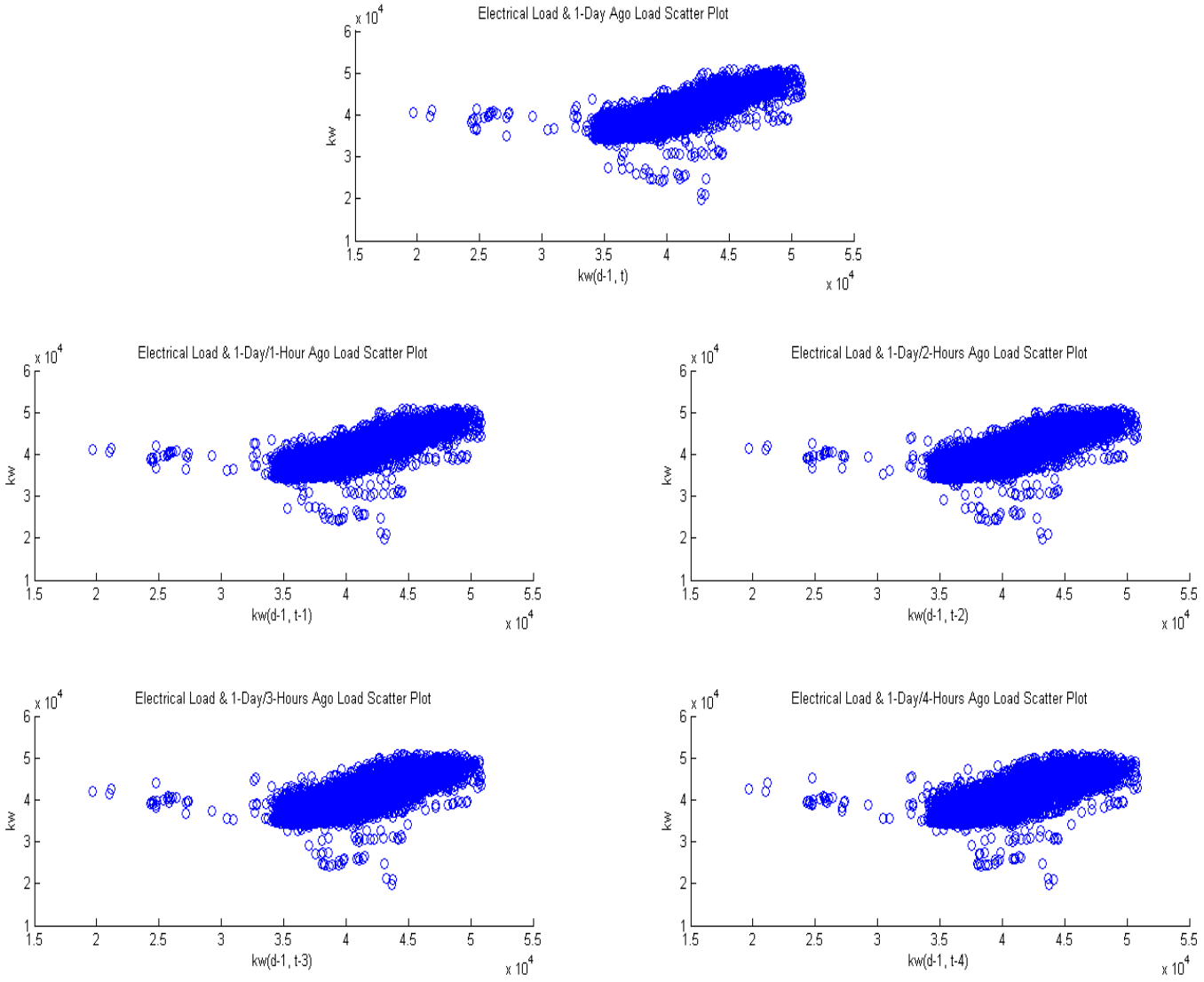


Figure 4-2. One year correlation between electric load and previous day's time-lagged loads.

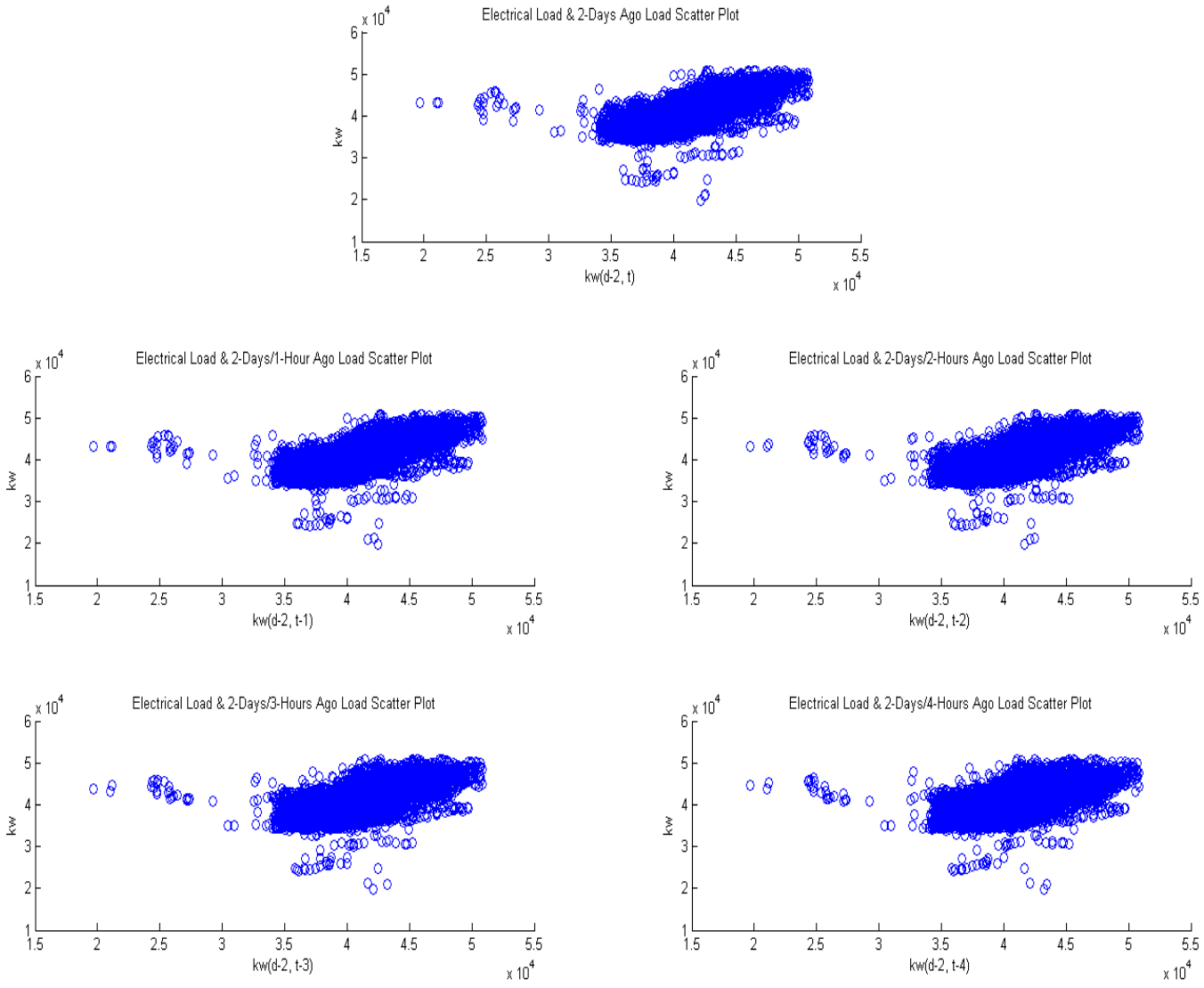


Figure 4-3. One year correlation between electric load and two days prior time-lagged loads.

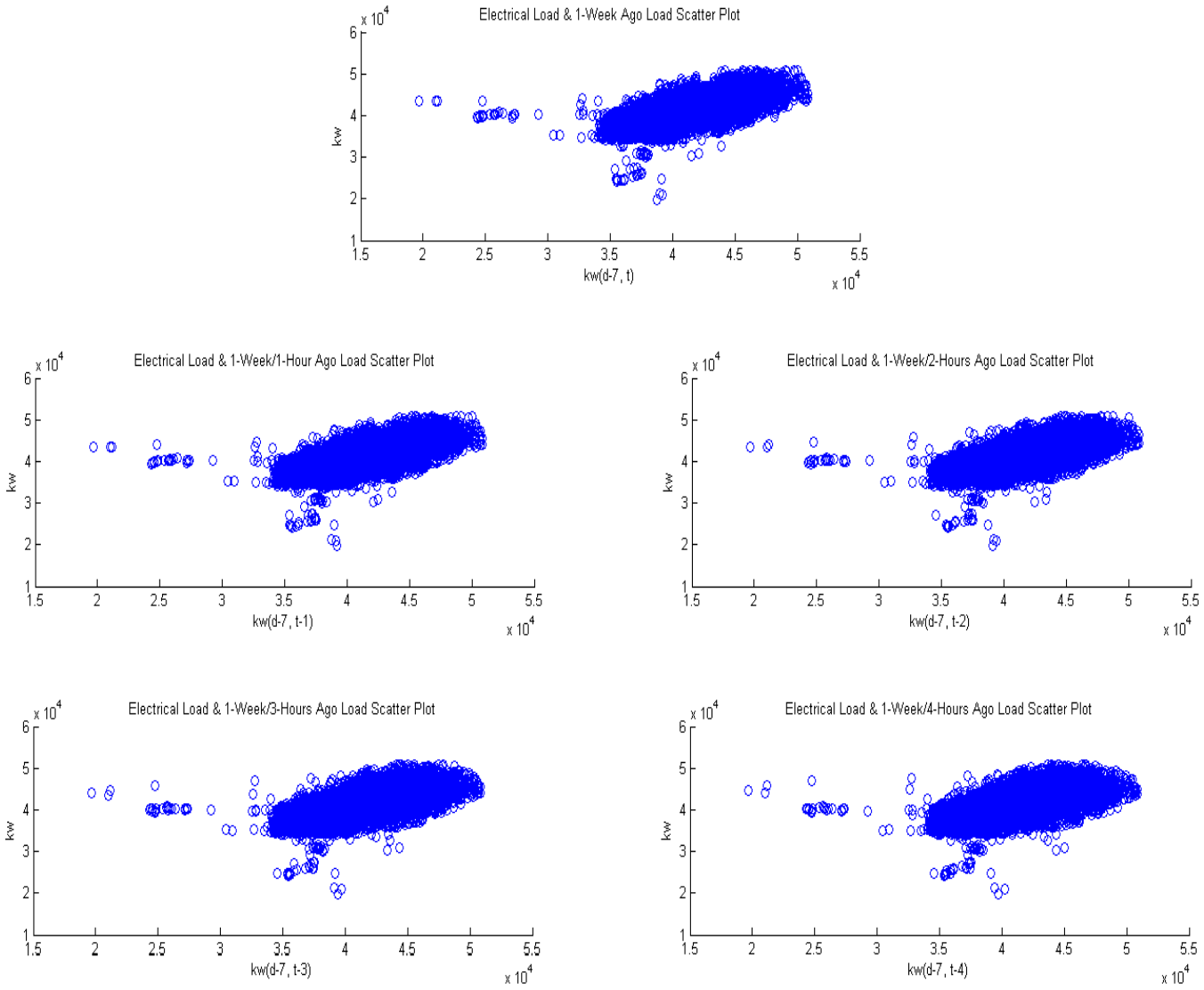


Figure 4-4. One year correlation between electric load and one-week time-lagged loads.

Fig. 4-2 shows a positive correlation between the forecast-time load and load values from the previous day, however, the correlations are not as strong as those in Fig. 4-1. The data point distribution in Fig. 4-2 is wider than Fig. 4-1, indicating the less correlated data. As in Fig. 4-1, as the time-lag in Fig. 4-2 increases, the correlation decreases.

Fig. 4-3 is the scatter plot of load and the two-days-ago load at the forecast time and the previous four hours.

Fig. 4-3 shows a positive correlation between the forecast-time load and load values from two days prior to the forecast day. The data set distributions in Fig. 4-3 are wider than those of Fig. 4-2 indicating less correlation with the forecast-time load.

Fig. 4-4 is the scatter plot of load and one-week-ago load at the forecast time and the previous four hours.

Fig. 4-4 shows a positive correlation between the forecast-time load and load values from one week prior to the forecast day. The data point distributions of Fig. 4-4 appear to have a more consistent form than those in Figs. 4-2 and 4-3. This more cohesive shape indicates the correlation of the one-week time-lagged loads is probably higher than the prior one-day and two-day time-lagged loads.

Fig. 4-5 shows the scatter plots of load versus air temperature, solar irradiance, relative humidity, atmospheric pressure, wind speed, and precipitation. The upper-left subplot shows a definite correlation between load and air temperature. When the air temperature is low, the load is high. As the air temperature increases, the load drops to

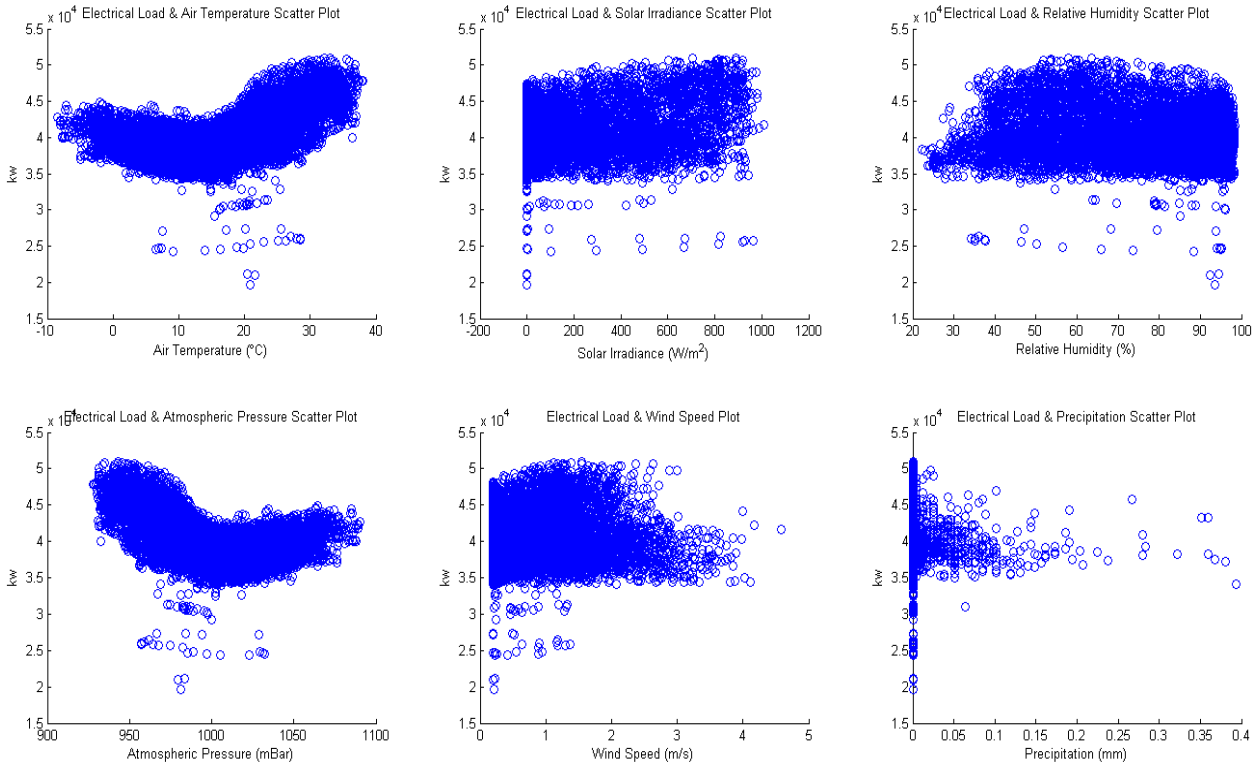


Figure 4-5. One year correlation between electric load and weather variables.

an average minimum at approximately 15 °C. As the air temperature continues to increase, the load rises to values higher than during the coldest temperatures. This subplot illustrates the temperature extremes for ORNL are approximately -8 °C and 38 °C. It also bounds the normal load range for ORNL from a minimum of approximately 34,000 kW to a maximum of approximately 51,000 kW. The outlier low load values are most likely attributable to planned or unplanned outages and holidays.

Fig. 4-5's upper-middle subplot shows the scatter plot of load and solar irradiance. Viewing the plot, it is difficult to discern any correlation between the two variables, however, this subplot illustrates the bounds of solar irradiance for ORNL between 0 and approximately 950 W/m^2.

Fig. 4-5's upper-right subplot shows the scatter plot of load and relative humidity. Like the load and solar irradiance subplot, it is difficult to discern any correlation between the load and relative humidity. This subplot does bound ORNL's minimum relative humidity to a range approximately 25 % to 35 % and a maximum of approximately 100 %.

Fig. 4-5's lower-left subplot shows the scatter plot of load and atmospheric pressure. The pressure is an estimate based on the air temperature and elevation of the weather station. Its dependence on temperature is apparent in that it appears to approximate a mirror image of the load and temperature's scatter plot. This subplot bounds ORNL's pressure to an approximate minimum of 930 mBar and an approximate maximum of 1080 mBar.

Fig. 4-5's lower-middle subplot shows the scatter plot of load and average wind speed. There does not appear to be any correlation between these two values. This subplot shows the typical bounds of ORNL's average wind speed which ranges from 0 m/s to approximately 2-5 m/s.

Fig. 4-5's lower-right subplot shows the scatter plot of load and hourly precipitation. There does not appear to be any significant correlation between the two values. This subplot bounds ORNL's typical hourly maximum precipitation to approximately 0.1 mm.

Fig. 4-6 is the scatter plot of load and calendar/time descriptors month, day of week, and time of day.

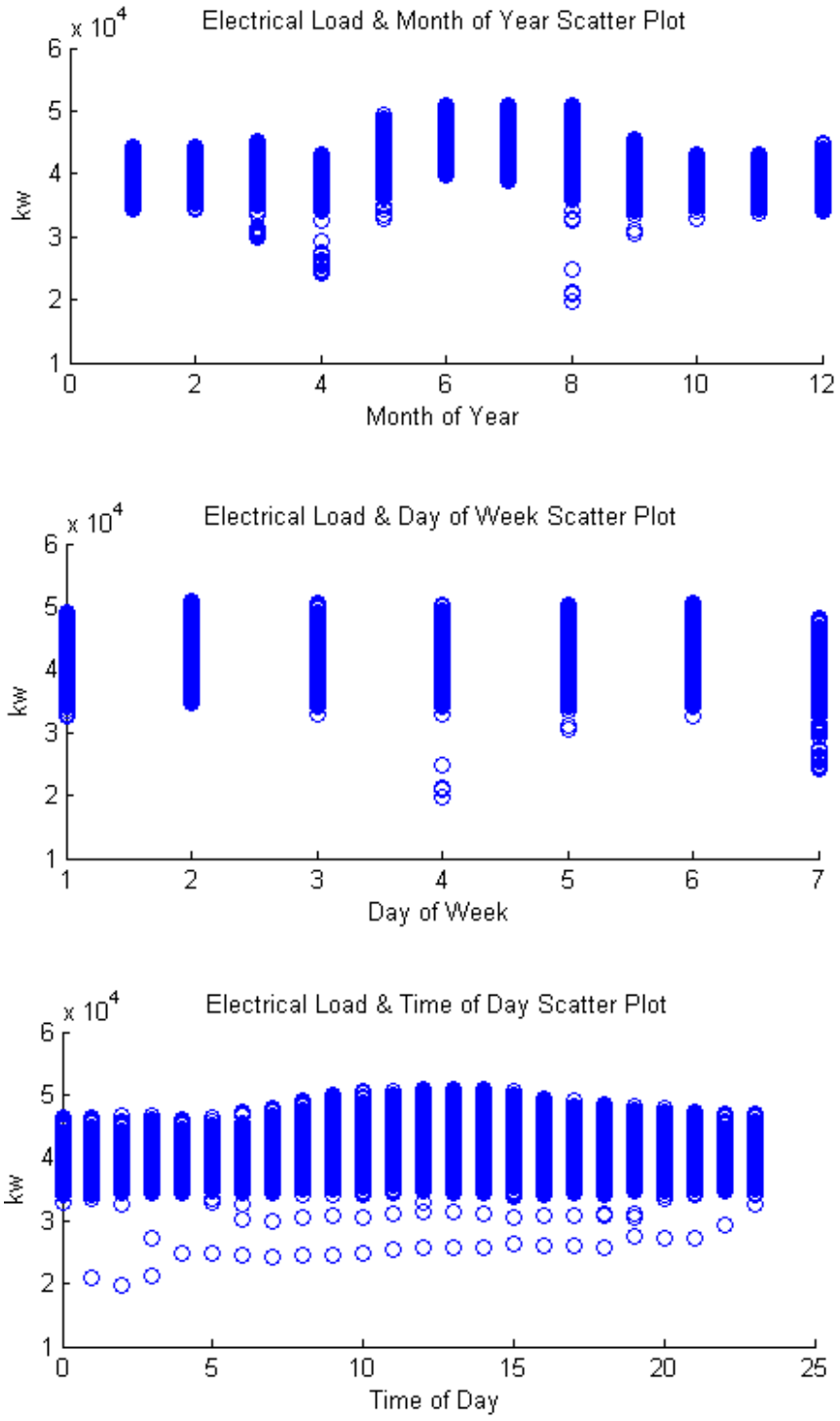


Figure 4-6. One year correlation between electric load and calendar/time descriptors.

The upper subplot of Fig. 4-6 illustrates the upper and lower bounds of load for each month of the year. The summer season (May-August) has higher maximums and minimum load values as compared to the other months of the year.

The middle subplot of Fig. 4-6 shows the minimum to maximum load range for each day of the week over a year. The load range for Sunday-Friday (Days 1-6) are very similar. The minimum load range for Saturday (Day 7) is lower than the other days. This may be due to planned outages being performed on the weekend to reduce the impact to lab operations during the work week.

The lower subplot of Fig. 4-6 illustrates the range of loads over each hour of the day. The plot has the highest load values between the hours of 0900 and 1500 hours. During this time range, the maximum load values of winter season peaks (which occur approximately from 0600 to 1000 hours) and spring, summer, and fall season peaks (which occur approximately from 1300 to 1700 hours) are blended, thus producing a higher bound in the plot during these times. It is of interest to note that other than for a few spurious low load values, ORNL's lower load values are bounded at approximately 33,000 kW, regardless of time of day.

4.2 Correlation Analysis

Pearson's correlation coefficient was utilized to determine the magnitude of linear correlation between load and the predictor values. The correlation analysis results for a Wednesday from each season are tabularized in Table 4-1.

Table 4-1 shows that the previous hour's load is highly correlated with the current hour's load. This correlation drops as the time-lag, t, is increased for the same day, d.

Table 4-1
Correlation analysis results.

Date	7/20/2011	10/5/2011	2/1/2012	3/14/2012
kW(d, t-1)	0.9577	0.9797	0.959	0.9546
kW(d, t-2)	0.8822	0.9412	0.9166	0.8957
kW(d, t-3)	0.7874	0.8909	0.8712	0.8298
kW(d, t-4)	0.6778	0.8326	0.8225	0.7593
kW(d-1, t)	0.7206	0.7254	0.6744	0.5845
kW(d-1, t-1)	0.6963	0.705	0.6643	0.5802
kW(d-1, t-2)	0.6437	0.67	0.6431	0.5574
kW(d-1, t-3)	0.5673	0.6237	0.6126	0.5294
kW(d-1, t-4)	0.473	0.5698	0.5835	0.4987
kW(d-2, t)	0.538	0.5065	0.4184	0.4765
kW(d-2, t-1)	0.5217	0.4937	0.4071	0.4822
kW(d-2, t-2)	0.4812	0.4677	0.3938	0.4752
kW(d-2, t-3)	0.4183	0.4311	0.3758	0.4643
kW(d-2, t-4)	0.338	0.3867	0.3591	0.4541
kW(d-7, t)	0.3263	0.6287	0.3795	0.414
kW(d-7, t-1)	0.3017	0.6205	0.3725	0.3988
kW(d-7, t-2)	0.2456	0.5984	0.3523	0.3637
kW(d-7, t-3)	0.1643	0.5651	0.3249	0.3231
kW(d-7, t-4)	0.0648	0.5232	0.2961	0.2798
Air Temperature	0.6512	0.7227	-0.3902	-0.4952
Irradiance	0.6466	0.434	-0.009	-0.0957
Humidity	-0.5224	-0.4109	0.0207	0.2718
Air Pressure	-0.6531	-0.7178	0.3931	0.5007
Precipitation	-0.0438	0.0246	-0.047	-0.0271
Wind Speed	0.47	0.1826	-0.0263	-0.1018
Month	0.1069	-0.4901	-0.5969	-0.3655
Day of Week	-0.0479	0.0923	-0.0615	0.0188
Hour	0.2084	0.0968	-0.1482	-0.1793

During the summer season, the correlation coefficient increases slightly at time t for the previous one- and two-day lags $d-1$ and $d-2$, but decreases from this value as the time-lag is increased during the respective day. During the fall season, the correlation coefficient increases slightly at time t for the seven-day (one week). A similar trend is evident during the winter season, only the increase in the correlation coefficient is less.

The air temperature correlation coefficient is higher for the summer and fall seasons indicating a higher correlation between load and air temperature during those seasons as compared to winter and spring. The air temperature correlation coefficient is negative for the winter seasons. This behavior is illustrated in Fig. 3-2 where the winter season load and temperature profiles appear out of phase with each other. The air temperature correlation coefficient for the spring date in Table 4-1 is also negative. This is a result of the transition from heating loads in the winter season to cooling loads in the spring season. Other dates in the spring season have positive air temperature coefficients. Fig. 3-3 indicates this transition occurs at approximately 15 °C.

Table 4-1 indicates a high correlation between load and solar irradiance during the summer season, a smaller correlation during the fall season, and relatively no correlation during the winter and spring season dates.

Table 4-1 shows the relative humidity and atmospheric pressure correlation coefficients are opposite in sign to the air temperature coefficients of the same season. This occurs because the humidity and pressure values are higher during the night hours.

The precipitation and average wind speed correlation coefficients are very small for all seasons. This is a result of the low rain fall and winds recorded during the training

period. Over a very small time span, these coefficients might be higher, but over the multiple weeks used for ANN training, these parameters have a smaller effect.

Table 4-1 shows the correlation coefficients for the calendar designators of Month, Day of Week, and Hour have various signs, but all have relatively low magnitudes. This illustrates that the Pearson's correlation coefficient does not perform well for nonlinear relationships. It is obvious from Fig. 4-6 that these parameters are correlated with the load.

The minimum correlation coefficient used for determining predictor inputs was 0.7. This value was selected by trial and error to determine which correlation coefficient value would produce the lowest forecast error.

4.3 Load Forecast Results

The 24-hour-ahead load forecast results for one week from each season are tabularized in Table 4-2.

The Maximum Percent Error column in Table 4-2 is the highest integer error value obtained for all hours from the 24-hour load forecast day.

The MAPE results in Table 4-2 have an approximate range of 1% to 3 %. These errors are comparable to the MAPE results found in the STLF literature [25]. The higher error values marked with an asterisk represent days where the actual load profile experienced a planned or unplanned outages or other sudden load change. This is drawback to using an ANN to forecast the load. The ANN cannot predict these events, so a one or two hour outage on a major distribution feeder will have a significant impact on the resulting error. If these events did not occur, then the error would have been much

Table 4-2
24-hour load forecast results.

Season	Date	Day of Week	MAPE (%)	Maximum Percent Error (%)	RMSE (kW)
Summer	7/17/2011	1	0.96	-2.32	496
	7/18/2011	2	1.22	-2.95	700
	7/19/2011	3	1.39	2.35	683
	7/20/2011	4	2.03	3.77	1127
	7/21/2011	5	2.6 *	-7.46	1501
	7/22/2011	6	0.93	2.74	567
	7/23/2011	7	1.75	3.15	905
Fall	10/2/2011	1	3.13 *	-8.71	1364
	10/3/2011	2	2.84 *	-6.07	1203
	10/4/2011	3	3.58 *	-13.65	1754
	10/5/2011	4	1.78	5.41	935
	10/6/2011	5	1.8	6.17	987
	10/7/2011	6	0.65	-1.59	306
	10/8/2011	7	0.74	1.83	317
Winter	1/29/2012	1	1.38	-5.5	718
	1/30/2012	2	1.76	5.28	976
	1/31/2012	3	1.41	-3.83	687
	2/1/2012	4	1.76	-3.96	866
	2/2/2012	5	1.3	3.42	641
	2/3/2012	6	1	-2.37	487
	2/4/2012	7	1.53	-3.96	701
Spring	3/11/2012	1	1.03	-2.46	485
	3/12/2012	2	1.64	-3.61	765
	3/13/2012	3	1.77	-4.55	808
	3/14/2012	4	3.93	7.27	1823
	3/15/2012	5	1.99	5.73	1085
	3/16/2012	6	3.51	-7.52	1608
	3/17/2012	7	1.54	-4.09	751

less. The higher error values for Day 4 (Wednesday) and Day 6 (Friday) of the spring season forecasts were the result of the ANN not anticipating the increase in load magnitude of those two days. The magnitude of the load trended higher for these two days than for the associated days in the training dataset.

Figs. 4-7 through 4-10 are plots showing the forecasted and actual load profiles for the Wednesdays in Table 4-2. These plots also show the absolute percent error profile over the 24-hour span.

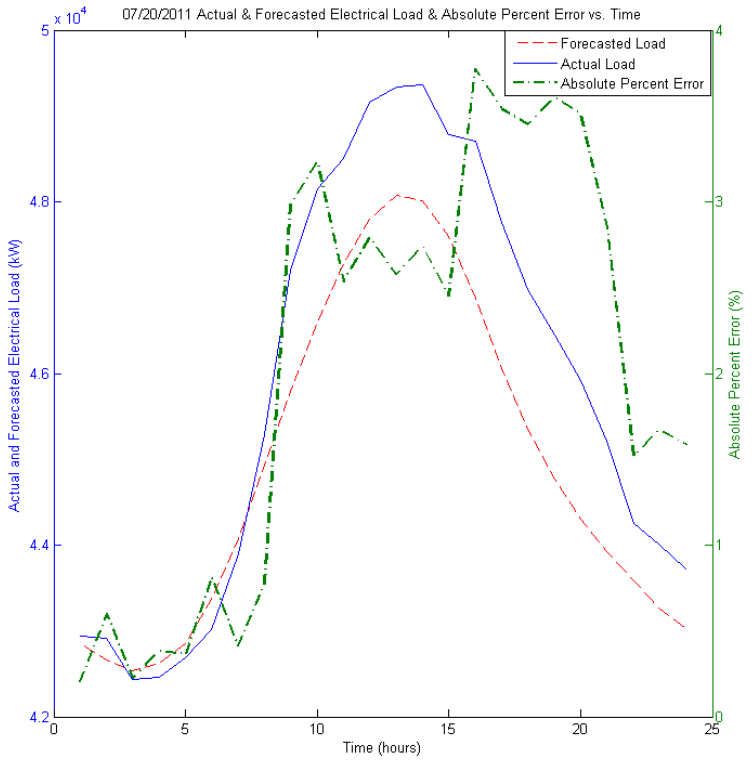


Figure 4-7. Forecasted and actual load profile for July 20, 2011.

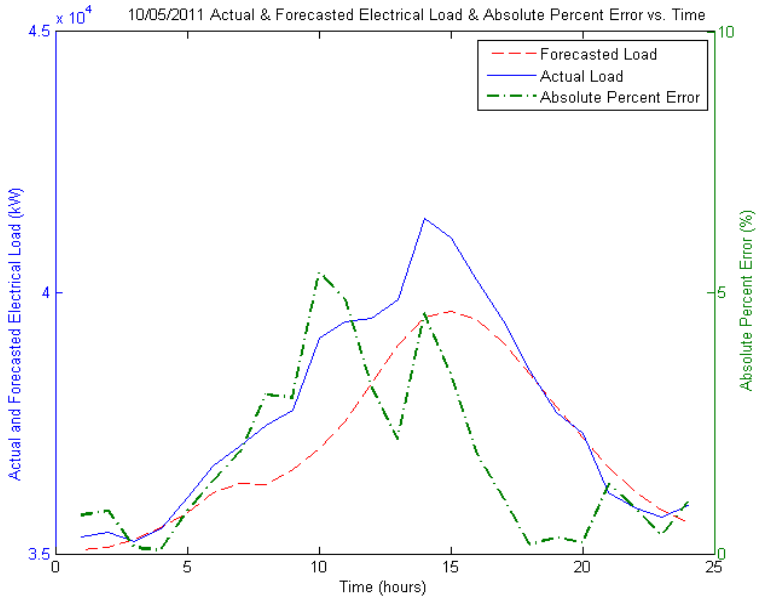


Figure 4-8. Forecasted and actual load profile for October 5, 2011.

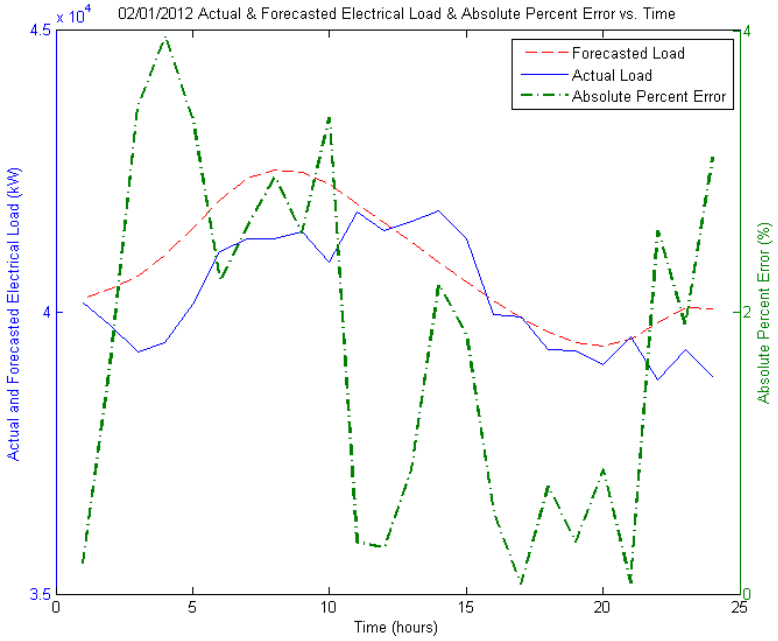


Figure 4-9. Forecasted and actual load profile for February 1, 2012.

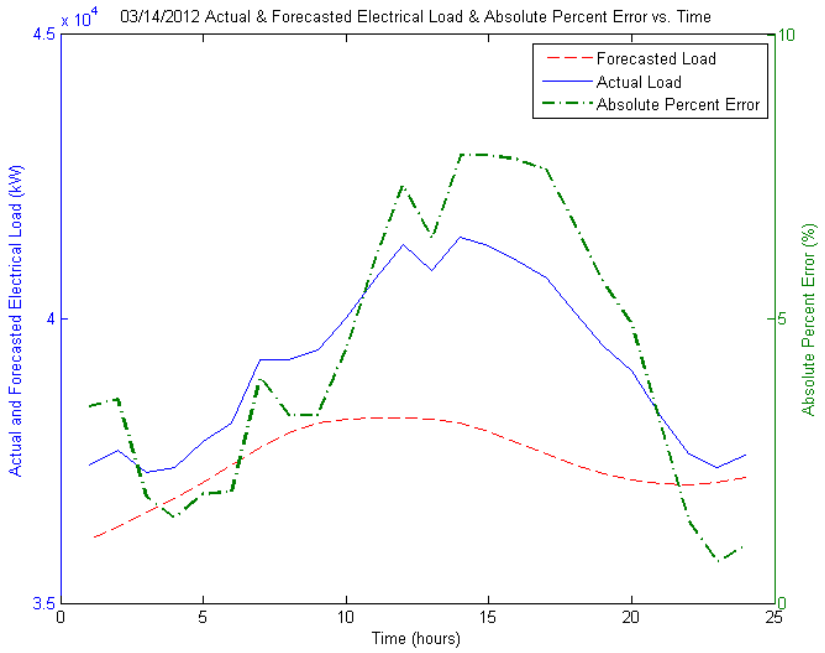


Figure 4-10. Forecasted and actual load profile for March 14, 2012.

Table 4-3 lists the hourly forecasted load, actual load, and absolute percent errors associated with each forecasted load profile in Figs 4-7 through 4-10.

Figs. 4-7 through 4-10 and Table 4-3 indicate the higher absolute percent errors occurred during the hours when the load profile approached the peak, peaked, and left the peak. The higher error values on Feb. 1, 2011 occurred during minimum load levels. This illustrates the chaotic load profiles experienced during the winter season. The winter load profiles are not as smooth as the profiles during the other seasons, so the ANN has a harder time forecasting the changes in the profile.

All ANNs were trained on six weeks of input data. This value was selected by trial and error to determine the minimum number of training weeks that would produce the lowest forecast error. Limiting the training input data to a small number of weeks

Table 4-3
24-hour forecast hourly results

	7/20/2011			10/5/2011			2/1/2012			3/14/2012		
Hour	Forecast Load (kW)	Actual Load (kW)	Absolute Percent Error (%)	Forecast Load (kW)	Actual Load (kW)	Absolute Percent Error (%)	Forecast Load (kW)	Actual Load (kW)	Absolute Percent Error (%)	Forecast Load (kW)	Actual Load (kW)	Absolute Percent Error (%)
1	42852	42940	0.2038	35068	35332	0.7458	40258	40170	0.2188	36129	37422	3.4558
2	42659	42916	0.5978	35116	35406	0.8182	40417	39766	1.6373	36322	37700	3.6544
3	42532	42436	0.2256	35270	35230	0.1148	40646	39288	3.4573	36496	37296	2.1438
4	42625	42462	0.3841	35504	35480	0.0686	41024	39462	3.959	36670	37396	1.9408
5	42846	42690	0.3646	35792	36086	0.8149	41501	40144	3.3792	36990	37850	2.2712
6	43363	43014	0.8119	36175	36690	1.4032	41991	41078	2.2227	37414	38178	2.0016
7	44079	43898	0.4112	36351	37070	1.9403	42379	41302	2.6076	37838	39288	3.6908
8	44930	45284	0.7823	36331	37472	3.0456	42526	41302	2.9647	38167	39288	2.8532
9	45790	47200	2.9872	36620	37750	2.9924	42489	41428	2.561	38352	39464	2.817
10	46600	48156	3.2317	37018	39134	5.4063	42284	40900	3.3849	38415	40018	4.0051
11	47280	48510	2.5361	37519	39436	4.8613	41936	41782	0.3679	38443	40698	5.5413
12	47792	49164	2.7903	38257	39514	3.1802	41591	41454	0.3306	38476	41302	6.8418
13	48071	49342	2.5751	38994	39866	2.1885	41240	41606	0.8793	38473	40850	5.8181
14	48013	49366	2.7403	39523	41430	4.6038	40888	41806	2.1952	38418	41428	7.2656
15	47593	48788	2.4493	39638	41050	3.4403	40541	41302	1.8437	38291	41280	7.241
16	46873	48712	3.7745	39483	40246	1.896	40197	39968	0.5725	38103	41026	7.1254
17	46063	47754	3.5409	39048	39464	1.0539	39889	39916	0.0677	37901	40724	6.9311
18	45353	46974	3.4515	38435	38506	0.1839	39638	39338	0.7629	37716	40118	5.9864
19	44769	46444	3.6058	37817	37700	0.3103	39458	39314	0.3656	37555	39514	4.9565
20	44304	45916	3.5117	37214	37296	0.2211	39406	39060	0.8857	37431	39084	4.2301
21	43920	45208	2.8496	36648	36162	1.3444	39532	39564	0.0815	37357	38302	2.4662
22	43580	44250	1.5148	36192	35884	0.858	39808	38808	2.5759	37344	37624	0.743
23	43262	43998	1.6736	35828	35708	0.3354	40086	39338	1.9021	37394	37396	0.0062
24	43028	43722	1.5864	35580	35934	0.9846	40064	38860	3.098	37501	37598	0.2585

keeps the data within the same season, and so the use of Pearson's correlation coefficient is justified since the weather input variables in a specific season had linear correlations with the load values in that same season.

The number of hidden layer neurons in each ANN was set at 15. This value was selected by trial and error to determine the minimum number of hidden-layer neurons that would produce the lowest forecast error.

The values selected for the minimum correlation coefficient, number of training weeks, and number of hidden-layer neurons had a direct affect on the Matlab® program's ANN training runtime. A low minimum correlation coefficient value would allow more input variables to be used in the ANN predictor matrices. This added complexity would increase the program runtime and would also present the ANN with so much input data that the resulting forecast error would increase. Increasing the training weeks also required the ANN to process a larger amount of data, which increased the program runtime. If the input data set was too large, the ANN could not generalize its output, and the resulting forecast error would increase. Increasing the number of hidden-layer neurons increased the program runtime because the weights and biases of each neuron have to be calculated and optimized during network training. An overly complex network could not generalize on the out-of-sample data set and would "over fit" the in-sample data.

The Matlab® program ANN training runtime typically ranged from 4-6 minutes when using a minimum correlation coefficient of 0.7, six training weeks, and 15 hidden-layer neurons.

The next chapter summarizes the research, methods, and results of this thesis. Future research and improvements to the STLF for smaller power systems will be proposed.

Chapter 5

Conclusions and Recommendations

5.1 Thesis Conclusions

It is a goal for every power system manager to have their power system operate efficiently, securely, and economically. To meet this goal, the behavior of their power system must be understood. Analysis of the system's normal operating bounds, response to customer demands, and reaction to weather events will provide insight on system loading. Short-term electric load forecasting can provide that insight for the following day to assist in making power system operational decisions.

In addition to energy charges ($/kWh), ORNL is subject to electric power demand charges ($/kW) from its serving utility. These demand charges can quickly add up to a high value during months with continued high electrical loads. The demand charges and associated system loading can be reduced by energy demand management techniques such as conservation, on-site generation, electric load-shedding, or implementing demand-response agreements with the serving utility. Energy demand management is a process where the energy use for a facility is planned and coordinated with the system's various load centers.

Peak load reduction is a significant component of energy demand management. Electric load forecasting can be used to assist in planning how much electricity must be reduced during system peaks to meet the goals of the energy demand management program. The amount of peak reduction may change each day due to the load forecast and facility operations. If the difference between the electric load forecast and the

desired maximum peak load is known, facility operators could use this information to schedule when their high electricity operations are to occur. For example, it was shown in Chapter 2 that the ORNL daily winter peak load occurs during the mid-morning hours. Where feasible, high electricity demand loads should be scheduled for operation during the winter afternoons. The daily summer peak load occurs during the mid-afternoon hours. Where feasible, high electricity demand loads should be scheduled for operation during the summer mornings.

Various load forecasting methods were reviewed in Chapter 2. Multiple linear regression, stochastic time series, general exponential smoothing, and state-space methods are all traditional STLF methods which use statistical relationships, mathematical equations, and historical information to generate a load forecast. Various forms of these methods have been used by utilities to assist in short-term operations planning. The most popular are the stochastic time series models. However, most of these traditional models were intended to model linear processes, and electric load is an inherently nonlinear process.

Knowledge-based expert systems use computer programs to act as expert system operators and forecast the load based on that operational knowledge and IF-THEN statements. An artificial neural network (ANN) is a mathematical model that mimics the decision-making processes of the human brain. The fundamental component of the ANN is the neuron. Neurons are programmed to behave similarly to the neurons in the brain by receiving inputs, processing those inputs, and producing an output. The neurons are connected together to form a network that can be used to solve nonlinear problems.

The ANN is well suited for load forecasting because the neurons are designed to receive a number of inputs (historical load information, weather forecast, etc.) and process them through a nonlinear activation function. The neurons of the connected network are trained on system-specific inputs so they can predict trends in the load based on the inputs of other variables. It should not be assumed that an ANN trained with inputs and targets for one power system will produce low error forecasts when used on a different system.

The structure and size of the ANN should only be as complex as required to produce acceptable forecast on out-of-sample data. Overly complex ANNs can overfit their training data producing very low error on in-sample input data, but high error on out-of-sample input data. Overfitting reduces the ANN's ability to generalize on data it has not been trained on. In addition, a complex ANN will have a high training runtime as there are more weights and biases to be estimated during the learning process.

One method for reducing the complexity of the ANN is by performing a correlation analysis on the predictor data prior to training the ANN. If only input variables with high correlation are used, the ANN will be less complex, and less likely to be confused with erroneous data that adds no value to producing a low error load forecast. Over a long time span, some of the input predictors have a nonlinear correlation with the load. If the training data is limited to a small time range, the previously nonlinear input predictor data now has a linear relationship to the load. Under these circumstances, a linear correlation analysis can be performed to reduce the input predictor data to only the most relevant data.

Historical time-lagged load data had correlation coefficients that decreased the farther removed from the forecast date and time. However, during some of the seasons, there were instances where the historical load from the prior week had a higher correlation coefficient than the load from two days prior to the forecast date and time. The weather input variable correlation coefficient values are also influenced by the change in seasons.

The ANNs created during this research resulted in mean absolute percent errors that ranged from approximately 1 % to 3 %. This error range is consistent with the errors presented in the STLF literature. Since most existing literature considers only transmission-level load forecasting, the resulting errors in this research should be viewed as very good. The transmission-level forecasts benefit from load aggregation across the larger transmission network. Load changes on the smaller ORNL power system will have a larger overall effect on the system load, potentially causing higher error forecasts.

One of the difficulties in applying ANNs to load forecasting occurs when there is an outage on the system or a significant event that causes the load to change such as the start-up or shutdown of a high demand load. The ANN cannot be trained to unplanned outages because there is no way to predict when these will occur. There is also difficulty in training the ANN to planned outages because these to not occur often, therefore the ANN will not be exposed to many of them during its training phase.

In this thesis, the weather predictor variables for the forecast day were considered 100 % accurate. This rarely, if ever, will be the case in a real time implementation of

these ANNs. Therefore, there will be weather forecast error that should be addressed and will likely cause additional error in the final load forecast.

5.2 Future Research

ANN load forecasts are considered black-box methods. They are trained on a data set, presented with a forecast dataset, and output a value. This method often provides low error results, but as previously noted, there are times when events occur that dramatically affect the actual load which the ANN cannot predict. One such event is the power outage.

Unplanned outages will still result in higher error forecasts, but planned outages could be used as input to the ANN. If an outage is planned, the estimated affected load could be supplied as a dedicated input to the ANN. It would be better for the outage value to be supplied to the ANN as an input than just subtracting the affected outage load from the ANN forecast. When a significant load change event occurs on a power system, this has the effect of changing the load pattern for the rest of the day. This effect is called load inertia. For example, if a large load is switched into a power system, the load profile will have a step change at the switching time. Throughout the rest of the day, the load profile will have higher values for longer time spans. In addition to the new large load, the system will experience additional losses, causing equipment to run hotter, and require additional cooling. The inertia of this load change will diminish as the system load decreases for the day. However, if the load change event was large enough, the load inertia might carry over to the following day which would affect that day's load forecast. The same would be true if a large load was switched off of the power system. The

remaining load profile for the day would be less than normal, but the difference in load levels would not be exact value of the load that was removed.

If a power system experiences these kinds of large load change events on a regular period, then the ANN training algorithm should be capable of predicting this inertia and load step changes. However, if these events do not occur regularly, then they should be supplied as part of the forecast predictor data set so the ANN can adjust its load forecast accordingly. Planned outages and other system operational data was not provided for the ORNL power system and so was not included in this research. Future research should attempt to include power system operational information as inputs to the ANN.

List of References

1. Y. Al-Rashid and L.D. Paarmann, “Short-term electric load forecasting using neural network models,” *Circuits and Syst.*, Ames, IA, 1996, pp. 1436-1439.

2. I. Moghram and S. Rahman, “Analysis and evaluation of five short-term load forecasting techniques,” *IEEE Trans. Power Syst.*, vol. 4, pp. 1484-1491, Nov. 1989.

3. G. Gross and F.D. Galiana, “Short-Term Load Forecasting,” *Proc. IEEE*, vol. 75, pp. 1558-1573, Dec. 1987.

4. T. Hong, M. Gui, M. Baran, and H.L. Willis, “Modeling and forecasting hourly electric load by multiple linear regression with interactions,” *Power and Energy Soc. General Meeting*, Minneapolis, MN, 2010, pp. 1-8.

5. P.J. Santos, A.G. Martins, and A.J. Pires, “Short-term load forecasting based on ANN applied to electrical distribution substations,” *Universities Power Engineering Conf.*, Bristol, UK, 2004, vol. 1, pp. 427-432.

6. T.G. Manohar and V.C. Veera Reddy, “Load forecasting by a novel technique using ANN,” *ARPN J. of Eng. And Appl. Sci.*, vol. 3, pp. 19-25, Apr. 2008.

7. N. Amral, C.S. Ozveren, and D. King, “Short term load forecasting using multiple linear regression,” *Universities Power Engineering Conference*, Brighton, 2007, pp. 1192-1198.

8. W.R. Christiaanse, “Short-term load forecasting using general exponential smoothing,” *IEEE Trans. Power App. and Syst.*, vol. PAS-90, pp. 900-911, Mar. 1971.

9. S. Rahman and R. Bhatnagar, “An expert system based algorithm for short term load forecast,” *IEEE Trans. Power Syst.*, vol. 3, pp. 392-399, May 1988.

10. H.S. Hippert, C.E. Perdreira, and R.C. Souza, “Neural networks for short-term load forecasting: a review and evaluation,” *IEEE Trans. Power Syst.*, vol. 16, pp. 44-55, Feb. 2001.

11. M. Ramezani, H. Falaghi, and M. Haghifam, “Short-term electric load forecasting using neural networks,” *Int. Conf. on Computer as a Tool*, Belgrade, 2005, pp. 1525-1528.

12. K.Y. Lee, Y.T. Cha, and J.H. Park, “Short-term load forecasting using an artificial neural network,” *IEEE Trans. Power Syst.*, vol. 7, pp. 124-132, Feb. 1992.

13. T. Senjyu, P. Mandal, K. Uezato, and T. Funabashi, “Next day load curve forecasting using recurrent neural network structure,” *Proc. Inst. Elect. Eng.*, vol. 151, pp. 388-394, May 2004.

14. Z.H. Osman, M.L. Awad, and T.K. Mahmoud, “Neural network based approach for short-term load forecasting,” *IEEE/PES Power Systems Conference and Exposition*, Seattle, WA, Mar. 15-18, 2009.

15. F. Liu, R.D. Findlay, and Q. Song, “A neural network based short term electric load forecasting in Ontario Canada,” *Int. Conf. on Computational Intelligence for Modeling, Control, and Automation*, Sydney, NSW, 2006, pp. 119-126.

16. W. Charytoniuk and M. Chen, “Very short-term load forecasting using artificial neural networks,” *IEEE Trans. Power Syst.*, vol. 15, pp. 263-268, Feb. 2000.

17. J.K. Mandal, A.K. Sinha, and G. Parthasarathy, “Application of recurrent neural network for short term load forecasting in electric power system,” *Int. Conf. on Neural Networks*, Perth, WA, 1995, pp. 2694-2698.

18. D.C. Park, M.A. El-Sharkawi, R.J. Marks II, L.E. Atlas, and M.J. Damborg, “Electric load forecasting using an artificial neural network,” *IEEE Trans. Power Syst.*, vol. 6, pp. 442-449, May 1991.

19. B.S. Kermanshahi et al, “Artificial neural network for forecasting daily loads of a canadian electric utility,” *Applications of Neural Networks to Power Systems*, Yokohama, Japan, 1993, pp. 302-307.

20. Y. Hsu and C. Yang, “Design of artificial neural networks for short-term load forecasting. Part II: Multiplayer feedforeward networks for peak load and valley load forecasting,” *Proc. Inst. Elect. Eng.*, vol. 138, pp. 414-418.

21. T. Senjyu, H. Takara, and K. Uezato, “Two-hour-ahead load forecasting using neural network,” *Int. Conf. on Power Syst. Technology*, Perth, WA, 2000, pp. 1119-1124.

22. T. Senjyu, H. Takara, K. Uezato, and T. Funabashi, “One-hour-ahead load forecasting using neural network,” *IEEE Trans. Power Syst.*, vol. 17, pp. 113-118, Feb. 2002.

23. H. Demuth, M. Beale, and M. Hagan. (2009, Mar.). *Neural network toolbox™ 6 user’s guide* [Online]. Available: http://www.varpa.org/Docencia/Files/nnet.pdf

24. Weisstein, Eric W. "Correlation Coefficient." *MathWorld*- [Online]. Available: http://mathworld.wolfram.com/CorrelationCoefficient.html

25. I. Drezga and S. Rahman, “Short-term load forecasting with local ANN predictors,” *IEEE Trans. Power Syst.*, vol. 14, pp. 844-850, Aug. 1999.

26. A.T. Sapeluk, C.S. Ozveren, and A.P. Birch, “Short term electric load forecast using artificial neural networks,” *Mediterranean Electrotechnical Conf.*, Antalya, 1994, pp. 905-908.

Appendix

Appendix A - Matlab® Code

Script “stlf.m”:

```
% Short-term Load Forecast (STLF) main file
% Eric Taylor, Graduate Student, University of Tennessee, 1/1/2013
% Run this file to implement STLF
% Edit function "input_data" as required when changing spreadsheet data
clear all
close all
time=cputime;   % records when program started


%% Load All Data %%
% load input data into Matlab workspace from Excel spreadsheets
% calls "input_data" function
[kw, kw_1hour, kw_2hour, kw_3hour, kw_4hour, kw_1day, kw_1day1hour, kw_1day2hour,
kw_1day3hour, kw_1day4hour, kw_2day, kw_2day1hour, kw_2day2hour, kw_2day3hour,
kw_2day4hour, kw_1week, kw_1week1hour, kw_1week2hour, kw_1week3hour, kw_1week4hour,
temperature, irradiance, humidity, pressure, precipitation, wind_speed, date, month,
day_of_week, hour, weekend, holiday]=input_data7();
data_length = length(kw);   % length of input data vectors (all the same length except
holiday_dates)


%% Defines Predictor Data for Preprocessing %%
x = [kw, kw_1hour, kw_2hour, kw_3hour, kw_4hour, kw_1day, kw_1day1hour, kw_1day2hour,
kw_1day3hour, kw_1day4hour, kw_2day, kw_2day1hour, kw_2day2hour, kw_2day3hour,
kw_2day4hour, kw_1week, kw_1week1hour, kw_1week2hour, kw_1week3hour, kw_1week4hour,
temperature, irradiance, humidity, pressure, precipitation, wind_speed];
x1 = [date, month, day_of_week, hour, weekend, holiday];


%% NN Forecast and Training Date Info %%
% % If forecast date info is not available, this calls "user_input" function
% % input_date: mm/dd/yyyy
% % input_weekend: 1 = weekend (Sunday or Saturday), 0 = weekday
% % input_holiday: 1 = holiday, 0 = not a holiday
forecast_date = '07/21/2011';
input_weekend = 'Y';   % not used
input_holiday = 'N';   % not used
if isempty(forecast_date)
    [input_date, input_weekend, input_holiday]=user_input();
else input_date = datenum(forecast_date,'mm/dd/yyyy');
end

% define start and finish dates for forecast and training
start_forecast = find(date==input_date,1);   % 1st data for forecast day
finish_forecast = start_forecast + 23;   % last data for 24-hour forecast day
weeks = 6;   % defines # of weeks for training set
start = start_forecast - (24*7*weeks);   % 1st data for training set
finish = start_forecast - 1;   % last data for training set


%% Input Data Correlation to Load Data %%
min_correlation = 0.7;   % defines minimum value of correlation of input data to load

% predictor matrix for training and forecast set
[prediction_train,prediction_forecast,data_type] =
correlation(x,x1,start,finish,min_correlation,start_forecast,finish_forecast);
prediction_length = length(prediction_train);


%% 24-Hour (Day-ahead) Forecast %%
% calculates 24-hour ahead electrical load forecast
max_neurons = 15;   % maximum # of hidden layer neurons
kw_actual = kw(start_forecast:finish_forecast);   % actual load data for forecast day
```

```
kw_train = kw(start:finish);   % load data for training set

% determines actual load data before forecast day (skips weekend data)
if day_of_week(start_forecast) == 2   % if forecast day is a Monday
    if holiday(start_forecast-24*3) == 1   % if Friday is a holiday
        kw_previous = kw((start_forecast-24*7):finish-24*6);   % actual load data for
Monday 1 week before forecast day of Monday
        prediction_previous = prediction_train((prediction_length-24*6-
23):prediction_length-24*6,:);   % predictor data for Monday 1 week before forecast day
of Monday
    else   % use Friday's predictor data
        kw_previous = kw((start_forecast-24*3):finish-24*2);   % actual load data for
Friday before forecast day of Monday
        prediction_previous = prediction_train((prediction_length-24*2-
23):prediction_length-24*2,:);   % predictor data for Friday before forecast day of
Monday
    end
else   % use previous day's predictor data
    kw_previous = kw((start_forecast-24):finish);   % actual load data for day before
forecast day
    prediction_previous = prediction_train((prediction_length-23):prediction_length,:);
% predictor data for day before forecast day
end

[day_ahead_forecast5, rmse5, max_perr5, ape5, mape5, day_ahead_forecast1, rmse1,
max_perr1, ape1, mape1, day_ahead_forecast, rmse, max_perr, ape,
mape]=NN_min_error2(prediction_train, prediction_forecast, prediction_previous, kw_train,
kw_actual, kw_previous, max_neurons, data_type);

figure(1)
plot(day_ahead_forecast,'r--')
hold on
[AX,H1,H2] = plotyy(1:24,kw_actual,1:24,ape);
set(get(AX(1),'Ylabel'),'String','Actual and Forecasted Electrical Load (kW)')
set(get(AX(2),'Ylabel'),'String','Absolute Percent Error (%)')
set(H2,'LineStyle','-.','LineWidth',2)
xlabel('Time (hours)')
legend('Forecasted Load','Actual Load','Absolute Percent Error');
title([forecast_date,' Actual & Forecasted Electrical Load & Absolute Percent Error vs.
Time'])
hold off

time=cputime-time;   % calcs program run time
time_in_minutes = time/60   % converts program run time to minutes
```

Function “input_data7.m”:

```
function[kw, kw_1hour, kw_2hour, kw_3hour, kw_4hour, kw_1day, kw_1day1hour, kw_1day2hour,
kw_1day3hour, kw_1day4hour, kw_2day, kw_2day1hour, kw_2day2hour, kw_2day3hour,
kw_2day4hour, kw_1week, kw_1week1hour, kw_1week2hour, kw_1week3hour, kw_1week4hour,
temperature, irradiance, humidity, pressure, precipitation, wind_speed, date, month,
day_of_week, hour, weekend, holiday]=input_data7()
% Input data creator for artificial neural network electric load forecaster
% Reads power, weather, and date data from Excel spreadsheets and returns
% the following vectors:
% kw, kw_1hour, kw_2hour, kw_3hour, kw_4hour, kw_1day, kw_1day1hour,
% kw_1day2hour, kw_1day3hour, kw_1day4hour, kw_2day, kw_2day1hour,
% kw_2day2hour, kw_2day3hour, kw_2day4hour, kw_1week, kw_1week1hour,
% kw_1week2hour, kw_1week3hour, kw_1week4hour, temperature, irradiance,
% humidity, pressure, precipitation, wind_speed, date, month, day_of_week,
% hour, weekend, holiday

% Eric Taylor, Graduate Student, University of Tennessee, 1/1/2013
% Note: A leap-year occurs in this data set, so there are 366 days in this
% one year's worth of data


%% Electrical Load Data %%
% import electrical load data from one Excel file (two worksheets)
% data is recorded in Daylight Savings Time (DST)
load_data_original_part1 = xlsread('ORNL_2011_2012.xlsx','FY11 Apr11-Sep11');
load_data_original_part2 = xlsread('ORNL_2011_2012.xlsx','FY12 Oct11-Mar12');
load_data_original = [load_data_original_part1; load_data_original_part2];

% converts Excel serial date format to Matlab serial date format
datecol = 7;   % input data column with serial date value
load_data_original(:,datecol) = load_data_original(:,datecol) + datenum('30-Dec-1899');

% sorting load data into component parts
load_data_size = size(load_data_original);
a=1;   % counter
% start for loop at 2 because even indices are half-hour load values
for n = 2:2:load_data_size   % step size is 2 because only want even value datapoints
    kw(a) = load_data_original(n,1);
    kw_losses(a) = load_data_original(n,2);
    kvar(a) = load_data_original(n,3);
    kvar_losses(a) = load_data_original(n,4);
    intervals_per_hour(a) = load_data_original(n,5);
    day_of_week(a) = load_data_original(n,6);
    date(a) = load_data_original(n,7);
    a = a + 1;
end;

kw = kw';
kw_losses = kw_losses';
kvar = kvar';
kvar_losses = kvar_losses';
intervals_per_hour = intervals_per_hour';
day_of_week = day_of_week';
date = date';


%% Time-lag %%
% creates time-lagged load data
% 1-4 hour lag for same day, 1 day lag, 2 day lag, & 1 week lag for each load sample
data_length = length(kw);   % length of input data vectors (they are all the same)
% same day delays
delay_1hour = 1;   % 1 hour delay
delay_2hour = 2;   % 2 hour delay
delay_3hour = 3;   % 3 hour delay
```

```
delay_4hour = 4;   % 4 hour delay
% 1 day delays
delay_1day = 1*24;   % 1 day delay (24 hrs)
delay_1day1hour = 1*24+1;   % 1 day & 1 hour delay (25 hrs)
delay_1day2hour = 1*24+2;   % 1 day & 2 hour delay (26 hrs)
delay_1day3hour = 1*24+3;   % 1 day & 3 hour delay (27 hrs)
delay_1day4hour = 1*24+4;   % 1 day & 4 hour delay (28 hrs)
% 2 day delays
delay_2day = 2*24;   % 2 day delay (48 hrs)
delay_2day1hour = 2*24+1;   % 2 day & 1 hour delay (49 hrs)
delay_2day2hour = 2*24+2;   % 2 day & 2 hour delay (50 hours)
delay_2day3hour = 2*24+3;   % 2 day & 3 hour delay (51 hours)
delay_2day4hour = 2*24+4;   % 2 day & 4 hour delay (52 hours)
% 1 week delays
delay_1week = 1*24*7;   % 1 week delay (168 hours)
delay_1week1hour = 1*24*7+1;   % 1 week & 1 hour delay (49 hrs)
delay_1week2hour = 1*24*7+2;   % 1 week & 2 hour delay (50 hours)
delay_1week3hour = 1*24*7+3;   % 1 week & 3 hour delay (51 hours)
delay_1week4hour = 1*24*7+4;   % 1 week & 4 hour delay (52 hours)

% initializes lag storage matrices and fills all elements with NaN
kw_1hour = NaN(data_length,1);
kw_2hour = NaN(data_length,1);
kw_3hour = NaN(data_length,1);
kw_4hour = NaN(data_length,1);
kw_1day = NaN(data_length,1);
kw_1day1hour = NaN(data_length,1);
kw_1day2hour = NaN(data_length,1);
kw_1day3hour = NaN(data_length,1);
kw_1day4hour = NaN(data_length,1);
kw_2day = NaN(data_length,1);
kw_2day1hour = NaN(data_length,1);
kw_2day2hour = NaN(data_length,1);
kw_2day3hour = NaN(data_length,1);
kw_2day4hour = NaN(data_length,1);
kw_1week = NaN(data_length,1);
kw_1week1hour = NaN(data_length,1);
kw_1week2hour = NaN(data_length,1);
kw_1week3hour = NaN(data_length,1);
kw_1week4hour = NaN(data_length,1);

% creates same day, various hour lag data
kw_1hour(1+delay_1hour:data_length) = kw(1:data_length-delay_1hour);
kw_2hour(1+delay_2hour:data_length) = kw(1:data_length-delay_2hour);
kw_3hour(1+delay_3hour:data_length) = kw(1:data_length-delay_3hour);
kw_4hour(1+delay_4hour:data_length) = kw(1:data_length-delay_4hour);
% creates 1 day, same and various hour lag data
kw_1day(1+delay_1day:data_length) = kw(1:data_length-delay_1day);
kw_1day1hour(1+delay_1day1hour:data_length) = kw(1:data_length-delay_1day1hour);
kw_1day2hour(1+delay_1day2hour:data_length) = kw(1:data_length-delay_1day2hour);
kw_1day3hour(1+delay_1day3hour:data_length) = kw(1:data_length-delay_1day3hour);
kw_1day4hour(1+delay_1day4hour:data_length) = kw(1:data_length-delay_1day4hour);
% creates 2 day, same and various hour lag data
kw_2day(1+delay_2day:data_length) = kw(1:data_length-delay_2day);
kw_2day1hour(1+delay_2day1hour:data_length) = kw(1:data_length-delay_2day1hour);
kw_2day2hour(1+delay_2day2hour:data_length) = kw(1:data_length-delay_2day2hour);
kw_2day3hour(1+delay_2day3hour:data_length) = kw(1:data_length-delay_2day3hour);
kw_2day4hour(1+delay_2day4hour:data_length) = kw(1:data_length-delay_2day4hour);
% creates 1 week, same and various hour lag data
kw_1week(1+delay_1week:data_length) = kw(1:data_length-delay_1week);
kw_1week1hour(1+delay_1week1hour:data_length) = kw(1:data_length-delay_1week1hour);
kw_1week2hour(1+delay_1week2hour:data_length) = kw(1:data_length-delay_1week2hour);
kw_1week3hour(1+delay_1week3hour:data_length) = kw(1:data_length-delay_1week3hour);
kw_1week4hour(1+delay_1week4hour:data_length) = kw(1:data_length-delay_1week4hour);
```

```
%% Month %%
% creates a matrix that stores only the month for each data point
month_str = datestr(date,'mm');
month = str2num(month_str);


%% Weekends %%
% determines which data occur on weekends (day_of_week = 1 or 7)
% 1 = weekend (Sunday or Saturday), 0 = weekday
weekend = zeros(data_length,1);
for a = 1:data_length
    if day_of_week(a) == 1   % Sunday
        weekend(a) = 1;
    elseif day_of_week(a) == 7   % Saturday
        weekend(a) = 1;
    else
    end
end


%% Weather Data %%
weather_data_original = xlsread('Weather_Data.xlsx');

% sorting weather data into component parts
% data is recorded in Standard Time
weather_data_size = size(weather_data_original);
hour = weather_data_original(:,1);
irradiance = weather_data_original(:,2);
temperature = weather_data_original(:,3);
humidity = weather_data_original(:,4);
wind_speed = weather_data_original(:,5);
pressure = weather_data_original(:,6);
precipitation = weather_data_original(:,7);


%% Holiday Data %%
% import holiday date data from one Excel file (two worksheets)
holiday_dates_part1 = xlsread('ORNL_holiday.xlsx','2011');
holiday_dates_part2 = xlsread('ORNL_holiday.xlsx','2012');
holiday_dates = [holiday_dates_part1; holiday_dates_part2];

% converts Excel serial date format to Matlab serial date format
datecol = 1;   % column with serial date value
holiday_dates(:,datecol) = holiday_dates(:,datecol) + datenum('30-Dec-1899');

% determines which data occur on ORNL holidays
% 1 = holiday, 0 = not a holiday
holiday = zeros(data_length,1);
holiday_dates_length = length(holiday_dates);
for a = 1:data_length
    for b = 1:holiday_dates_length
        if date(a) == holiday_dates(b)
            holiday(a) = 1;
        else
        end
    end
end
```

Function “user_input.m”:

```
function[input_date, input_weekend, input_holiday]=user_input()
% Forecast date creator for artificial neural network electric load forecaster
% Prompts user to input forecast date in mm/dd/yyyy format
% and if date is a weekend (Y/N) and/or a holiday (Y/N).
% Returns the following: input_date, input_weekend, & input_holiday

% Eric Taylor, Graduate Student, University of Tennessee, 1/1/2013


%% User Input for Forecast Date
% User inputs mm/dd/yyyy of date for STLF
a=0;
while a == 0
    input_date_str = input('Enter date for Short-term Load Forecast (mm/dd/yyyy): ','s');
    if isempty(input_date_str)
        disp('Invalid entry. Press any key to try again');
        pause;
    elseif length(input_date_str) ~= 10
        disp('Invalid date format. Press any key to try again');
        pause;
    elseif input_date_str(3) ~= '/'
        disp('Invalid date format. Press any key to try again');
        pause;
    elseif input_date_str(6) ~= '/'
        disp('Invalid date format. Press any key to try again');
        pause;
    else
        % converts date from a string to a serial number
        input_date = datenum(input_date_str,'mm/dd/yyyy');
        a = 1;
    end
end

% User inputs if date is a weekend (Y/N)
% 1 = weekend (Sunday or Saturday), 0 = weekday
a=0;
while a == 0
    input_weekend_str = input('Is date a weekend (Y/N)?: ','s');
    input_weekend_str = upper(input_weekend_str);   % converts input to uppercase
    if input_weekend_str == 'Y'
        input_weekend = 1;
        a = 1;
    elseif input_weekend_str == 'N'
        input_weekend = 0;
        a = 1;
    else
        disp('Invalid entry. Press any key to try again');
        pause;
    end
end

% User inputs if date is a holiday (Y/N)
% 1 = holiday, 0 = not a holiday
a=0;
while a == 0
    input_holiday_str = input('Is date a holiday (Y/N)?: ','s');
    input_holiday_str = upper(input_holiday_str);   % converts input to uppercase
    if input_holiday_str == 'Y'
        input_holiday = 1;
        a = 1;
    elseif input_holiday_str == 'N'
        input_holiday = 0;
        a = 1;
```

```
    else
        disp('Invalid entry. Press any key to try again');
        pause;
    end
end
```

Function “correlation.m”:

```
function [prediction_train,prediction_forecast,data_type] =
correlation(x,x1,start,finish,min_correlation,start_forecast,finish_forecast)
% finds input parameters that meet minimum correlation threshold
% x = [kw, kw_1hour, kw_2hour, kw_3hour, kw_4hour, kw_1day, kw_1day1hour, kw_1day2hour,
kw_1day3hour, kw_1day4hour, kw_2day, kw_2day1hour, kw_2day2hour, kw_2day3hour,
kw_2day4hour, kw_1week, kw_1week1hour, kw_1week2hour, kw_1week3hour, kw_1week4hour,
temperature, irradiance, humidity, pressure, precipitation, wind_speed]
% x1 = [date, month, day_of_week, hour, weekend, holiday]

data_type_x = {'kw', 'kw_1hour', 'kw_2hour', 'kw_3hour', 'kw_4hour', 'kw_1day',
'kw_1day1hour', 'kw_1day2hour', 'kw_1day3hour', 'kw_1day4hour', 'kw_2day',
'kw_2day1hour', 'kw_2day2hour', 'kw_2day3hour', 'kw_2day4hour', 'kw_1week',
'kw_1week1hour', 'kw_1week2hour', 'kw_1week3hour', 'kw_1week4hour', 'temperature',
'irradiance', 'humidity', 'pressure', 'precipitation', 'wind_speed'};

% correlation study
corr_results = corrcoef(x(start:finish,:));   % correlation study results matrix; only
need to use 1st column

% finds indices of inputs in "x" matrix that meet the minimum correlation threshold
min_corr_index = find(abs(corr_results(:,1))>=min_correlation);   % don't use 1st row;
it's autocorrelation (kw vs. kw)

columns = length(min_corr_index)-1;   % defines number of columns in prediction matrices
prediction_train1 = zeros(finish-start+1,columns);   % initializes to zero
prediction_forecast1 = zeros(finish_forecast-start_forecast+1,columns);   % initializes
to zero

% builds predictor matrices for training and forecast set using load and weather data
for a = 1:columns
    prediction_train1(:,a) = x(start:finish,min_corr_index(a+1));   % 1st row of
corr_results is autocorrelation (kw vs. kw); don't use
    prediction_forecast1(:,a) = x(start_forecast:finish_forecast,min_corr_index(a+1));
end
data_type1 = data_type_x(min_corr_index((1:columns)+1));

% adds day of week, hour, & weekend data from x1 to predictor matrices
% day of week, hour, & weekend data usually have poor linear correlation, but there
% is an obvious correlation between this data and load (see scatter plots)
prediction_train = [prediction_train1,x1(start:finish,3:5)];
prediction_forecast = [prediction_forecast1,x1(start_forecast:finish_forecast,3:5)];
data_type = [data_type1,'Day of Week','Hour','Weekend'];   % stores complete list of type
of input data used in prediction matrices

% adds Holiday to predictors if a holiday occurs during the training set
q=0;   % state variable
for m=start:finish   % training set range
    if x1(m,6)==1
        q=q+1;   % change state
    end
end
if q ~= 0
    prediction_train = [prediction_train,x1(start:finish,6)];
    prediction_forecast = [prediction_forecast,x1(start_forecast:finish_forecast,6)];
    data_type = [data_type,'Holiday'];
end
```

Function “NN_min_error2.m”:

```
function[day_ahead_forecast5, rmse5, max_perr5, ape5, mape5, day_ahead_forecast1, rmse1,
max_perr1, ape1, mape1, day_ahead_forecast, rmse, max_perr, ape,
mape]=NN_min_error2(prediction_train, prediction_forecast, prediction_previous, kw_train,
kw_actual, kw_previous, max_neurons, data_type)
% Trains NN a number of times, selects the NN with lowest error for a set
% range of neurons

% Eric Taylor, Graduate Student, University of Tennessee, 2/11/2013
max_iter=10;
train_iter=5;   % number of training iterations to determine minimum error
day_ahead_forecast_sum = zeros(24,1);   % initializes to zero
day_ahead_forecast5_sum = zeros(24,1);   % initializes to zero

neurons=max_neurons;  % number of neurons in hidden layer
for i = 1:max_iter
    for b = 1:train_iter
        % creates and trains NN
        [net_train] = NN_creator(prediction_train, kw_train, neurons);
        % tests NN with actual data
        day_ahead = 0;
        [kw_forecast_train] = forecast(net_train, prediction_train, day_ahead);
        % compute the mean absolute percent error from trained NN
        err = kw_train-kw_forecast_train;   % error
        perr_train = abs(err./kw_train*100);   % percent error
        mape_train = nanmean(perr_train);   % mean absolute percent error

        % 24-hour ahead forecast
        day_ahead = 1;
        [previous_day_forecast_train]=forecast(net_train, prediction_previous, day_ahead,
data_type);
        perr_previous_train = abs((kw_previous-
previous_day_forecast_train)./kw_previous*100);
        mape_previous_train = nanmean(perr_previous_train);

        [day_ahead_forecast_train]=forecast(net_train, prediction_forecast, day_ahead,
data_type);

        % minimize error
        if b == 1 % 1st iteration is initially stored as min error
            mape_store = mape_train;
            day_ahead_forecast_store = day_ahead_forecast_train;

            net5_store = net_train;
            mape_previous_store = mape_previous_train;
        else
            if mape_train < mape_store   % determines if new error is smaller than
previous minimum error
                mape_store = mape_train;
                day_ahead_forecast_store = day_ahead_forecast_train;
            end
            if mape_previous_train < mape_previous_store
                net5_store = net_train;
                mape_previous_store = mape_previous_train;
            end
        end
    end

    day_ahead = 1;
    [day_ahead_forecast5_store]=forecast(net5_store, prediction_forecast, day_ahead,
data_type);
    day_ahead_forecast5_sum = day_ahead_forecast5_sum + day_ahead_forecast5_store;
```

```
    day_ahead_forecast_sum = day_ahead_forecast_sum + day_ahead_forecast_store;   %
stores sum of day ahead forecasts
end


%% 24-hour forecast minimized using training mape
day_ahead_forecast1 = day_ahead_forecast_sum/max_iter;
err1 = kw_actual - day_ahead_forecast1;   % error
rmse1 = sqrt(nanmean((err1).^2));   % root mean square error
perr1 = (err1./kw_actual)*100;   % percent error
max_perr1 = perr1(find(abs(perr1)==max(abs(perr1))));   % maximum percent error
ape1 = abs(perr1);   % absolute percent error
mape1 = nanmean(ape1);   % mean absolute percent error


%% 24-hour forecast minimized using previous day's mape
day_ahead_forecast5 = day_ahead_forecast5_sum/max_iter;
err5 = kw_actual - day_ahead_forecast5;   % error
rmse5 = sqrt(nanmean((err5).^2));   % root mean square error
perr5 = (err5./kw_actual)*100;   % percent error
max_perr5 = perr5(find(abs(perr5)==max(abs(perr5))));   % maximum percent error
ape5 = abs(perr5);   % absolute percent error
mape5 = nanmean(ape5);   % mean absolute percent error


%% Choose which forecast result to use
previous_mape1 = mean(abs((kw_previous-day_ahead_forecast1)./kw_previous*100));   % mape
based on previous day load
previous_mape5 = mean(abs((kw_previous-day_ahead_forecast5)./kw_previous*100));   % mape
based on previous day load
if previous_mape1 <= previous_mape5
    day_ahead_forecast = day_ahead_forecast1;
    rmse = rmse1;   % root mean square error
    max_perr = max_perr1;   % maximum percent error
    ape = ape1;   % absolute percent error
    mape = mape1;   % mean absolute percent error
else
    day_ahead_forecast = day_ahead_forecast5;
    rmse = rmse5;   % root mean square error
    max_perr = max_perr5;   % maximum percent error
    ape = ape5;   % absolute percent error
    mape = mape5;   % mean absolute percent error
end
```

Function “NN_creator.m”:

```
function[net]=NN_creator(x, y, neurons)
%Function calculates 24-hour ahead electrical load forecast

% Solve an Input-Output Fitting problem with a Neural Network
% This script assumes these variables are defined:
%   x - input data.
%   y - target data.

inputs = x';
targets = y';

% Create a Fitting Network
hiddenLayerSize = neurons;
net = fitnet(hiddenLayerSize);

% Choose Input and Output Pre/Post-Processing Functions
net.inputs{1}.processFcns = {'removeconstantrows','mapminmax'};
net.outputs{2}.processFcns = {'removeconstantrows','mapminmax'};

% Setup Division of Data for Training, Validation, Testing
net.divideFcn = 'dividerand';  % Divide data randomly
net.divideMode = 'sample';  % Divide up every sample
net.divideParam.trainRatio = 90/100;   % 90% training
net.divideParam.valRatio = 5/100;      % 5% validation
net.divideParam.testRatio = 5/100;     % 5% testing

% Training function
net.trainFcn = 'trainlm';  % Levenberg-Marquardt

% Performance Function
net.performFcn = 'mse';  % Mean absolute error

% Train the Network
[net,tr] = train(net,inputs,targets);
```

Function “forecast.m”:

```
function[kw_forecast]=forecast(net, prediction, day_ahead, data_type)
if day_ahead == 0
    kw_forecast = sim(net,prediction')';   % uses NN to make full data set kW forecast
else
    kw_forecast = zeros(24,1);
    for a = 1:24   % 24 hours in a day
        kw_forecast(a) = sim(net,prediction(a,:)')';   % uses NN to make 24-hour ahead kW
forecast
        if a < 24   % doesn't make the final update as prediction is only 24 elements
long
            for b = 1:length(data_type)
                if strcmp(char(data_type(b)),'kw_1hour')
                    prediction(a+1,b) = kw_forecast(a);   % updates prediction matrix
with previous hour's kW forecast
                elseif strcmp(char(data_type(b)),'kw_2hour') & a > 1
                    prediction(a+1,b) = kw_forecast(a-1);   % updates prediction matrix
with 2 hour ago kW forecast
                elseif strcmp(char(data_type(b)),'kw_3hour') & a > 2
                    prediction(a+1,b) = kw_forecast(a-2);   % updates prediction matrix
with 3 hour ago kW forecast
                elseif strcmp(char(data_type(b)),'kw_4hour') & a > 3
                    prediction(a+1,b) = kw_forecast(a-3);   % updates prediction matrix
with 4 hour ago kW forecast
                end
            end
        end
    end
end
```

Appendix B - ORNL Holiday Schedule

ORNL holiday schedule.

HOLIDAY	DATE	EXCEL SERIAL DATE
New Year's Day	Friday, December 31, 2010	40543
Martin Luther King's Birthday	Monday, January 17, 2011	40560
Good Friday	Friday, April 22, 2011	40655
Memorial Day	Monday, May 30, 2011	40693
Independence Day	Monday, July 04, 2011	40728
Independence Day companion	Tuesday, July 05, 2011	40729
Labor Day	Monday, September 05, 2011	40791
Thanksgiving Day	Thursday, November 24, 2011	40871
Thanksgiving companion	Friday, November 25, 2011	40872
Christmas companion	Friday, December 23, 2011	40900
Christmas Day	Monday, December 26, 2011	40903
New Year's Day	Monday, January 02, 2012	40910
Martin Luther King's Birthday	Monday, January 16, 2012	40924
Good Friday	Friday, April 06, 2012	41005
Memorial Day	Monday, May 28, 2012	41057
Independence Day	Wednesday, July 04, 2012	41094
Independence Day companion	Thursday, July 05, 2012	41095
Labor Day	Monday, September 03, 2012	41155
Thanksgiving Day	Thursday, November 22, 2012	41235
Thanksgiving companion	Friday, November 23, 2012	41236
Christmas companion	Monday, December 24, 2012	41267
Christmas Day	Tuesday, December 25, 2012	41268

Vita

Eric Lynn Taylor received his Bachelors of Science degree in Electrical Engineering from the University of Tennessee in May 2004. Eric is a licensed professional engineer in the state of Tennessee. For the past nine years, he has performed the duties of electrical design engineer for the Y-12 National Security Complex in Oak Ridge, TN. His responsibilities include planning and performing the electrical design for projects located in the Y-12 National Security Complex. His duties also include providing support during construction and start-up activities for these projects. His interests include distribution system protection and design.